This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☑ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.

OIPE
APR 08 2002
PATENT & TRADEMARK OFFICE

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 942 605 A2

EUROPEAN PATENT APPLICATION

(43) Date of publication:
15.09.1999 Bulletin 1999/37

(51) Int. Cl.[6]: **H04N 7/26**

(21) Application number: **99301795.3**

(22) Date of filing: **10.03.1999**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **10.03.1998 JP 05811898**
05.06.1998 JP 15724398

(71) Applicant: SONY CORPORATION
Tokyo 141 (JP)

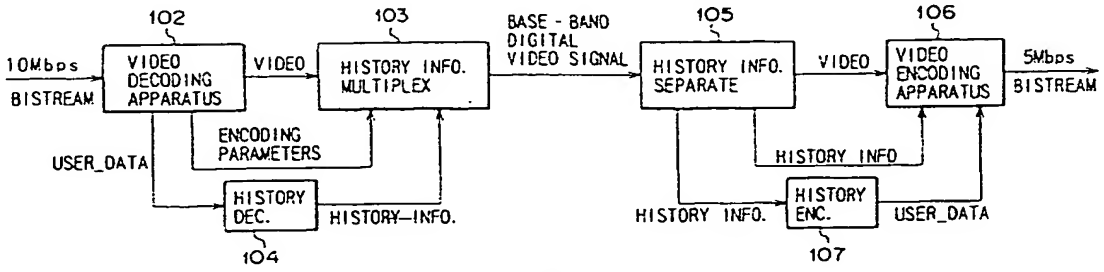
(72) Inventors:
- Tahara, Katsumi,
Intellectual Property Department
Shinagawa-ku, Tokyo 141 (JP)
- **Murakami, Yoshihiro,**
Intellectual Property Dep.
Shinagawa-ku, Tokyo 141 (JP)
- **Kitamura, Takuya,**
Intellectual Property Dep.
Shinagawa-ku, Tokyo 141 (JP)
- **Mihara, Kanji,**
Intellectual Property Dep.
Shinagawa-ku, Tokyo 141 (JP)

(74) Representative:
Turner, James Arthur et al
D. Young & Co.,
21 New Fetter Lane
London EC4A 1DA (GB)

(54) Transcoding system

(57) The present invention provides a transcoder for changing a GOP structure and the bit rate of an encoded bitstream obtained as a result of an encoding process based on MPEG standards. According to the transcoder provided by the present invention, encoding parameters generated in a past encoding process can be transmitted to the MPEG encoder which performs a present encoding process as a history information. And optimum encoding parameters in commensurate with the present encoding process are selected from the transmitted encoding parameters, and the selected encoding parameters are reused in the present encoding process. As a result, the picture quality does not deteriorate even if decoding and encoding processes are carried out repeatedly.

FIG. 14



EP 0 942 605 A2

Description

[0001] In recent years, the broadcasting station for producing and broadcasting television programs has been generally using an MPEG technology for compressing and encoding video data. In particular, the MPEG technology is becoming a de-facto standard for recording video data onto a tape or a random-accessible recording medium and for transmitting video data through a cable or a satellite.

[0002] The following is a brief description of typical processing carried out by broadcasting stations up to transmission of a video program produced in the station to each home. First, an encoder employed in a camcoder, (an apparatus integrating a video camera and a VTR into a single body), encodes source video data and records the encoded data onto a magnetic tape of the VTR. At that time, the encoder employed in the camcoder encodes the source video data into an encoded bitstream suitable for a recording format of the magnetic tape of the VTR. Typically, the GOP structure of an MPEG bitstream recorded on the magnetic tape is a structure wherein one GOP is composed of two frames. An example of the GOP structure is a structure comprising a sequence of pictures of the types of I-, B-, I-, B-, I-, B- and so on. The bit rate of the MPEG bitstream recorded on the magnetic tape is 18 Mbps.

[0003] Then, a central broadcasting station carries out edit processing to edit the video bitstream recorded on the magnetic tape. For this purpose, the GOP structure of the video bitstream recorded on the magnetic tape is converted into a GOP structure suitable for the edit processing. A GOP structure suitable for edit processing is a structure wherein one GOP is composed of one frame. To be more specific, pictures of a GOP structure suitable for edit processing are all I-pictures. This is because in order to carry out edit processing in frame units, the I-picture having no correlation with other pictures is most suitable. In the actual operation to convert the GOP structure, the video bitstream recorded on the magnetic tape is once decoded back into a base-band video data. Then, the base-band video data is re-encoded so as to comprise all I-pictures. By carrying out the decoding and re-encoding processes in this way, it is possible to generate a bitstream having a GOP structure suitable for edit processing.

[0004] Subsequently, in order to transmit an edited video program obtained as a result of the edit processing from the central broadcasting station to a local broadcasting station, it is necessary to change the GOP structure and the bit rate of the bitstream of the edited video program to a GOP structure and a bit rate that are suitable for the transmission. The GOP structure suitable for transmission between broadcasting stations is a GOP structure wherein one GOP is composed of 15 frames. An example of such a GOP structure is a structure comprising a sequence of pictures of the types of I-, B-, B-, P-, B-, B-, P- and so on. As for the bit rate suitable for transmission between broadcasting stations, a high bit rate of at least 50 Mbps is desirable since, in general, a dedicated line having a high transmission capacity such as an optical fiber is installed between broadcasting stations. To put it concretely, the bitstream of a video program completed the edit processing is once decoded back into a base-band video data. Then, the base-band video data is re-encoded to result in a GOP structure and a bit rate suitable for transmission between broadcasting stations as described above.

[0005] At the local broadcasting station, the video program received from the central broadcasting station is typically subjected to edit processing to insert commercials peculiar to the district where the local broadcasting station is located. Much like the edit processing carried out at the central broadcasting station, the bitstream of the video program received from the central broadcasting station is once decoded back into a base-band video data. Then, the base-band video data is encoded so as to comprise all I-pictures. As a result, it is possible to generate a bitstream having a GOP structure suitable for edit processing.

[0006] Subsequently, in order to transmit the video program completing the edit processing at the local broadcasting station to each home through a cable or a satellite, the GOP structure and the bit rate of the bitstream are converted into respectively a GOP structure and a bit rate that are suitable for the transmission to each home. A GOP structure suitable for the transmission to each home is a structure wherein one GOP is composed of 15 frames. An example of such a GOP structure is a structure comprising a sequence of pictures of the types of I-, B-, B-, P-, B-, B-, P- and so on. A bit rate suitable for transmission to each home has a typical value of as low as about 5 Mbps. The bitstream of a video program often completing the edit processing is decoded back into a base-band video data. Then, the base-band video data is re-encoded into a GOP structure and a bit rate suitable for transmission to each home.

[0007] As is obvious from the above description, a video program transmitted from the central broadcasting station to each home is subjected to repetitive decoding and encoding processes for a plurality of times during the transmission. In actuality, various kinds of signal processing other than the signal processing described above are carried out at a broadcasting station and the decoding and encoding processes are usually carried out for each kind of signal processing. As a result, the decoding and encoding processes need to be carried out repeatedly.

[0008] However, encoding and decoding processes based on MPEG standard are not 100 percent reverse processed to each other as is generally known. To be more specific, base-band video data subjected to an encoding process is not entirely the same as video data obtained as a result of a decoding process carried out in transcoding of the previous generation. Therefore,

decoding and encoding processes cause the picture quality to deteriorate. As a result, there is a problem of deterioration of the picture quality that occurs each time decoding and encoding processes are carried out. In other words, the effects of deterioration of the picture quality are accumulated each time decoding and encoding processes are repeated.

[0009] Various respective aspects and features of the invention are defined in the appended claims.

[0010] Embodiments of the present invention, addressing the problem described above, can provide a transcoding system, a video encoding apparatus, a stream procecessing system and a video decoding apparatus which cause no deterioration of the picture quality even if encoding and decoding processes are carried out repeatedly on a bitstream completing an encoding process based on MPEG standard in order to change the GOP structure and the bit rate of the bitstream. In addition, embodiments of the present invention can provide a transcoding system, a video encoding apparatus, a stream procecessing system and a video decoding apparatus causing no deterioration of the picture quality even if encoding and decoding processes are carried out repeatedly.

[0011] In embodiments of the invention, encoding parameters generated and used in a previous encoding process can be utilized in the current encoding process. As a result, the picture quality does not deteriorate even if decoding and encoding processes are carried out repeatedly. That is to say, it is possible to lessen the accumulated deterioration in the quality of picture due to repetition of the encoding process.

[0012] According to the transcoder provided by embodiments of the present invention, encoding parameters generated and used in a previous encoding process are described in a user-data area of an encoded bitstream obtained as a result of the current encoding process and the encoded bitstream conforms to MPEG standard. It is thus possible to decode the encoded bitstream by means of any existing decoders. In addition, it is not necessary to provide a dedicated line for transmitting encoding parameters generated and used in a previous encoding process. As a result, it is possible to transmit encoding parameters generated and used in a previous encoding process by utilizing the existing datastream transmission environment.

[0013] According to the transcoder provided by embodiments of the present invention, only selected previous encoding parameters generated and used in a encoding process are described in a user-data area of an encoded bitstream obtained as a result of the current encoding process. As a result, it is possible to transmit the encoding parameters generated and used in a encoding process carried out in the past without the need to substantially increase the bit rate of the output bitstream.

[0014] According to the transcoder provided by embodiments of the present invention, only encoding parameters optimum for the current encoding process are selected from encoding parameters generated and used in a previous encoding process to be used in the current encoding process. As a result, deterioration in the quality of picture is accumulated even if the previous decoding and encoding processes are carried out repeatedly.

[0015] According to the transcoder provided by embodiments of the present invention, only encoding parameters optimum for the current encoding process are selected in accordance with the picture types from previous encoding parameters generated and used in a previous encoding process to be used in the current encoding process. As a result, deterioration of the quality in picture is by no means accumulated even if decoding and encoding processes are carried out repeatedly.

[0016] According to the transcoder provided by embodiments of the present invention, a decision as to whether or not to reutilize previous encoding parameters generated and used in a previous encoding process is made on the basis of picture types included in the previous encoding parameters. As a result, an optimum encoding process can be carried out.

[0017] The invention will now be described by way of example with reference to the accompanying drawings, throughout which like parts are referred to by like references, and in which:

Fig. 1 is an explanatory diagram used for describing the principle of a high-efficiency encoding process;
Fig. 2 is an explanatory diagram showing picture types used ink compression of picture data;
Fig. 3 is an explanatory diagram showing picture types used in compression of picture data;
Fig. 4 is an explanatory diagram used for describing the principle of a process of encoding a moving-picture video signal;
Fig. 5 is a block diagram showing the configuration of an apparatus used for encoding and decoding a moving-picture video signal;
Figs. 6A to 6C are explanatory diagrams used for describing format conversion;
Fig. 7 is a block diagram showing the configuration of an encoder 18 employed in the apparatus shown in Fig. 5;
Fig. 8 is an explanatory diagram used for describing the operation of a prediction-mode switching circuit 52 employed in the encoder 18 shown in Fig. 7;
Fig. 9 is an explanatory diagram used for describing the operation of a prediction-mode switching circuit 52 employed in the encoder 18 shown in Fig. 7;
Fig. 10 is an explanatory diagram used for describing the operation of a prediction-mode switching circuit 52 employed in the encoder 18 shown in Fig. 7;
Fig. 11 is an explanatory diagram used for describing the operation of a prediction-mode switching circuit 52 employed in the encoder 18 shown in Fig. 7;
Fig. 12 is a block diagram showing the configuration

of a decoder 31 employed in the apparatus shown in Fig. 5;

Fig. 13 is an explanatory diagram used for describing SNR control based on picture types;

Fig. 14 is a block diagram showing the configuration of a transcoder 101 provided by the present invention;

Fig. 15 is a block diagram showing a more detailed configuration of the transcoder 101 shown in Fig. 14;

Fig. 16 is a block diagram showing the configuration of a decoder 111 employed in a decoding apparatus 102 of the transcoder 101 shown in Fig. 14;

Fig. 17 is an explanatory diagram showing pixels of a macroblock;

Fig. 18 is an explanatory diagram showing areas for recording encoding parameters;

Fig. 19 is a block diagram showing the configuration of an encoder 121 employed in an encoding apparatus 106 of the transcoder 101 shown in Fig. 14;

Fig. 20 is a block diagram showing a typical configuration of a history formatter 211 employed in the transcoder 101 shown in Fig. 15;

Fig. 21 is a block diagram showing a typical configuration of a history decoder 203 employed in the transcoder 101 shown in Fig. 15;

Fig. 22 is a block diagram showing a typical configuration of a converter 212 employed in the transcoder 101 shown in Fig. 15;

Fig. 23 is a block diagram showing a typical configuration of a stuff circuit 323 employed in the converter 212 shown in Fig. 22;

Figs. 24A to I are timing charts used for explaining the operation of the converter 212 shown in Fig. 22;

Fig. 25 is a block diagram showing a typical configuration of a converter 202 employed in the transcoder 101 shown in Fig. 15;

Figs 26 is a block diagram showing a typical configuration of a delete circuit 343 employed in the converter 202 shown in Fig. 25;

Fig. 27 is a block diagram showing another typical configuration of the converter 212 employed in the transcoder 101 shown in Fig. 15;

Fig. 28 is a block diagram showing another typical configuration of the converter 202 employed in the transcoder 101 shown in Fig. 15;

Fig. 29 is a block diagram showing a typical configuration of a user-data formatter 213 employed in the transcoder 101 shown in Fig. 15;

Fig. 30 is a block diagram showing the configuration of an actual system employing a plurality of transcoders 101 each shown in Fig. 14;

Fig. 31 is a diagram showing areas for recording encoding parameters;

Fig. 32 is a flowchart used for explaining processing carried out by the encoding apparatus 106 employed in the transcoder 101 shown in Fig. 14 to determine changeable picture types;

Fig. 33 is a diagram showing an example of changing picture types;

Fig. 34 is a diagram showing another example of changing picture types;

Fig. 35 is an explanatory diagram used for describing quantization control processing carried out by the encoding apparatus 106 employed in the transcoder 101 shown in Fig. 14;

Fig. 36 is a flowchart used for explaining quantization control processing carried out by the encoding apparatus 106 employed in the transcoder 101 shown in Fig. 14;

Fig. 37 is a block diagram showing the configuration of a tightly coupled transcoder 101;

Fig. 38 is an explanatory diagram used for describing the syntax of an MPEG stream;

Fig. 39 is an explanatory diagram used for describing the configuration of the syntax shown in Fig. 38;

Fig. 40 is an explanatory diagram used for describing the syntax of history_stream() for recording history information with a fixed length;

Fig. 41 is an explanatory diagram used for describing the syntax of history_stream() for recording history information with a fixed length;

Fig. 42 is an explanatory diagram used for describing the syntax of history_stream() for recording history information with a fixed length;

Fig. 43 is an explanatory diagram used for describing the syntax of history_stream() for recording history information with a fixed length;

Fig. 44 is an explanatory diagram used for describing the syntax of history_stream() for recording history information with a fixed length;

Fig. 45 is an explanatory diagram used for describing the syntax of history_stream() for recording history information with a fixed length;

Fig. 46 is an explanatory diagram used for describing the syntax of history_stream() for recording history information with a fixed length;

Fig. 47 is an explanatory diagram used for describing the syntax of history_stream() for recording history information with a variable length;

Fig. 48 is an explanatory diagram used for describing the syntax of sequence_header();

Fig. 49 is an explanatory diagram used for describing the syntax of sequence_extension();

Fig. 50 is an explanatory diagram used for describing the syntax of extension_and_user_data();

Fig. 51 is an explanatory diagram used for describing the syntax of user_data();

Fig. 52 is an explanatory diagram used for describing the syntax of group_of_picture_header();

Fig. 53 is an explanatory diagram used for describing the syntax of picture_header();

Fig. 54 is an explanatory diagram used for describing the syntax of picture_coding_extension();

Fig. 55 is an explanatory diagram used for describing the syntax of extension_data();

Fig. 56 is an explanatory diagram used for describing the syntax of quant_matrix_extension();
Fig. 57 is an explanatory diagram used for describing the syntax of copyright_extension();
Fig. 58 is an explanatory diagram used for describing the syntax of picture_display_extension();
Fig. 59 is an explanatory diagram used for describing the syntax of picture_data();
Fig. 60 is an explanatory diagram used for describing the syntax of slice();
Fig. 61 is an explanatory diagram used for describing the syntax of macroblock();
Fig. 62 is an explanatory diagram used for describing the syntax of macroblock_modes();
Fig. 63 is an explanatory diagram used for describing the syntax of motion_vectors(s);
Fig. 64 is an explanatory diagram used for describing the syntax of motion_vector(r, s);
Fig. 65 is an explanatory diagram used for describing a variable-length code of macroblock_type for an I-picture;
Fig. 66 is an explanatory diagram used for describing a variable-length code of macroblock_type for a P-picture: and
Fig. 67 is an explanatory diagram used for describing a variable-length code of macroblock_type for a B-picture.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0018] Before describing a transcoder provided by the present embodiments, a process to compress and encode a moving-picture video signal is explained. It should be noted that a technical term "system' used in this specification means a whole system comprising a plurality of apparatuses and means.

[0019] As described above, in a system for transmitting a moving-picture video signal to a remote destination such as a television-conference system and a television-telephone system, the video signal is subjected to compression and encoding processes using line correlation and intra-frame correlation of the video signal in order to allow the transmission line to be utilized with a high degree of efficiency. By using line correlation, a video signal can be compressed by carrying out typically DCT (Discrete Cosine Transform) processing.

[0020] By using intra-frame correlation, the video signal can be further compressed and encoded. Assume that frame pictures PC1, PC2 and PC3 are generated at points of time t1, t2 and t3 respectively as shown in Fig. 1. In this case, a difference in picture signal between the frame pictures PC1 and PC2 is computed to generate a frame picture PC12. By the same token, a difference in picture signal between the frame pictures PC2 and PC3 is computed to generate a frame picture PC23. Normally, a difference in picture signal between frame pictures adjacent to each other along the time axis is small. Thus, the amounts of information contained in the frame pictures PC21 and PC23 are small and the amount of code included in a difference signal obtained as a result of coding such a difference is also small.

[0021] By merely transmitting a difference signal, however, the original picture can not be restored. In order to obtain the original picture, frame pictures are classified into three types, namely, I-, P- and B-pictures each used as a smallest processing unit in the compression and encoding processes of a video signal.

[0022] Assume a GOP (Group of Pictures) of Fig. 2 comprising seventeen frames, namely, frames F1 to F17 which are each processed as a smallest unit of video signals for processing. To be more specific, the first frame F1, the second frame F2 and the third frame F3 are processed as I-, B- and P-pictures respectively. The subsequent frames, that is, the fourth to seventeenth frames F4 to F17, are processed as B- and P-pictures alternately.

[0023] In the case of an I-picture, a video signal of the entire frame is transmitted. In the case of a P-picture or a B-picture, on the other hand, only a difference in video signal can be transmitted as an alternative to the video signal of the entire frame. To be more specific, in the case of the third frame F3 of a P-picture shown in Fig. 2, only a difference in video signal between the P-picture and a chronically preceding I- or P-picture is transmitted as a video signal. In the case of the second frame F2 of a B-picture shown in Fig. 3, for example, a difference in video signal between the B-picture and a chronically preceding frame, a succeeding frame or an average value of the preceding and a chronically succeeding frames is transmitted as a video signal.

[0024] Fig. 4 is a diagram showing the principle of a technique of encoding a moving-picture video signal in accordance with what is described above. As shown in Fig. 4, the first frame F1 is processed as an I-picture. Thus, the video signal of the entire frame F1 is transmitted to the transmission line as data F1X (intra-picture encoding). On the other hand, the second frame F2 is processed as a B-picture. In this case, a difference between the second frame F2 and the chronically preceding frame F1, the succeeding frame F3 or an average value of the preceding frame F1 and the succeeding frame F3 is transmitted as data F2X.

[0025] To put it in detail, the processing of a B-picture can be classified into four types. In the processing of the first type, the data of the original frame F2 denoted by notation SP1 in Fig. 4 is transmitted as it is as data F2X as is the same case with an I-picture. Thus, the processing of the first type is thus the so-called intra-picture encoding. In the processing of the second type, a difference denoted by notation SP2 between the second frame F2 and the chronically succeeding third frame F3 is transmitted as data F2X. Since the succeeding frame is taken as a reference or a prediction picture, this processing is referred to as backward pre-

diction encoding. In the processing of the third type, a difference denoted by notation SP3 between the second frame F2 and the preceding first frame F1 is transmitted as data F2X as is the case with a P-picture. Since the preceding frame is taken as a prediction picture, this processing is referred to as forward prediction encoding. In the processing of the fourth type, a difference denoted by notation SP4 between the second frame F2 and an average of the succeeding third frame F3 and the preceding first frame F1 is transmitted as data F2X. Since the preceding and succeeding frames are taken as a prediction picture, this processing is referred to as forward & backward prediction encoding. In actuality, one of the four processing types aforementioned is selected so as to generate a minimum amount of transmission data obtained as a result of the processing.

[0026] It should be noted that, in the case of a difference obtained as a result the processing of the second, third or fourth type described above, a motion vector between the pictures of the frames (prediction picture) used in the computation of the difference is also transmitted along the difference. To be mote specific, in the case of the forward prediction encoding, the motion vector is a motion vector x1 between the frames F1 and F2. In the case of the backward prediction encoding, the motion vector is a motion vector x2 between the frames F2 and F3. In the case of the forward & backward prediction encoding, the motion vectors x1 and x2 are both transmitted.

[0027] Much like the B-picture described above, in the case of the frame F3 of the P-picture, the forward prediction encoding or the intra-picture processing is selected to produce a minimum amount of transmitting data obtained as a result of the processing. If the forward prediction encoding is selected, a difference denoted by notation SP3 between the third frame F3 and the preceding first frame F1 is transmitted as data F3X along with a motion vector x3. If the intra-picture processing is selected, on the other hand, it is the data F3X of the original frame F3 denoted by notation SP1 that is transmitted.

[0028] Fig. 5 is a block diagram showing a typical configuration of a system based on the principle described above to code a moving-picture video signal and transmit as well as decode the coded signal. A signal encoding apparatus 1 encodes an input video signal and transmits the encoded video signal to a signal decoding apparatus 2 through a recording medium 3 which serves as a transmission line. The signal decoding apparatus 2 plays back the coded signal recorded on the recording medium 3 and decodes the playback signal in to an output signal.

[0029] In the signal encoding apparatus 1, the input video signal is supplied to a preprocessing circuit 11 for splitting it into luminance and chrominance signals. In the case of this embodiment, the chrominance signal is a color-difference signal. The analog luminance and color-difference signals are then supplied to A/D converters 12 and 13 respectively to be each converted into a digital video signal. Digital video signals resulting from the A/D conversion are then supplied to a frame-memory unit 14 to be stored therein. The frame-memory unit 14 comprises a luminance-signal frame memory 15 for storing the luminance signal and a color-difference-signal frame memory 16 for storing the color-difference signal.

[0030] A format converting circuit 17 converts the frame-format signals stored in the frame-memory unit 14 into a block-format signal as shown in Figs. 6A to 6C. To put it in detail, a video signal is stored in the frame-memory unit 14 as data of the frame format shown in Fig. 6A. As shown in Fig. 6A, the frame format is a collection of V lines each comprising H dots. The format converting circuit 17 divides the signal of one frame into N slices each comprising 16 lines as shown in Fig. 6B. Each slice is then divided into M pieces of macroblocks as shown in Fig. 6B. As shown in Fig. 6C, a macroblock includes a luminance signal Y corresponding to 16 × 16 pixels (dots). The luminance signal Y is further divided into blocks Y[1] to Y[4] each comprising 8 × 8 dots. The 16×16-dot luminance signal is associated with a 8 × 8-dot Cb signal and a 8 × 8-dot Cr signal.

[0031] The data with the block format obtained as a result of the format conversion carried out by the format converting circuit 17 as described above is supplied to an encoder 18 for encoding the data. The configuration of the encoder 18 will be described later in detail by referring to Fig. 7.

[0032] A signal obtained as a result of the encoding carried out by the encoder 18 is output to a transmission line as a bitstream. Typically, the encoded signal is supplied to a recording circuit 19 for recording the encoded signal onto a recording medium 3 used as a transmission line as a digital signal.

[0033] A playback circuit 30 employed in the signal decoding apparatus 2 reproduces data from the recording medium 3, supplying the data to a decoder 31 of the decoding apparatus for decoding the data. The configuration of the decoder 31 will be described later in detail by referring to Fig. 12.

[0034] Data obtained as a result of the decoding carried out by the decoder 31 is supplied to a format converting circuit 32 for converting the block format of the data back into a frame format. Then, a luminance signal having a frame format is supplied to a luminance-signal frame memory 34 of a frame-memory unit 33 to be stored therein. On the other hand, a color-difference signal having a frame format is supplied to a color-difference-signal frame memory 35 of the frame-memory unit 33 to be stored therein. The luminance signal is read back from the luminance-signal frame memory 34 and supplied to a D/A converter 36. On the other hand, the color-difference signal is read back from the color-difference-signal frame memory 35 and supplied to a D/A converter 37. The D/A converters 36 and 37 convert the signals into analog signals which are then supplied to a

post-processing circuit 38 for synthesizing the luminance and color-difference signals and generating a synthesized output.

[0035] Next, the configuration of the encoder 18 is described by referring to Fig. 7. Picture data to be encoded is supplied to a motion-vector detecting circuit 50 in macroblock units. The motion-vector detecting circuit 50 processes picture data of each frame as an I-, P- or B-picture in accordance with a predetermined sequence set in advance. To be more specific, picture data of a GOP typically comprising frames the F1 to F17 as shown in Figs. 2 and 3 is processed as a sequence of I-, B-, P-, B-, P-, ---, B- and P-pictures.

[0036] Picture data of a frame to be processed by the motion-vector detecting circuit 50 as an I-picture such as the frame F1 shown in Fig. 3 is supplied to a forward-source-picture area 51a of a frame-memory unit 51 to be stored therein. Picture data of a frame to be processed by the motion-vector detecting circuit 50 as a P-picture such as the frame F2 is supplied to a referenced-source-picture area 51b of the frame-memory unit 51 to be stored therein. Picture data of a frame processed by the motion-vector detecting circuit 50 as a P-picture such as the frame F3 is supplied to a backward-source-picture area 51c of the frame-memory unit 51 to be stored therein.

[0037] When picture data of the next two frames such as the frame F4 or F5 are supplied sequentially to the motion-vector detecting circuit 50 to be processed as B and P-pictures respectively, the areas 51a, 51b and 51c are updated as follows. When picture data of the frame F4 is processed by the motion-vector detecting circuit 50, the picture data of the frame F3 stored in the backward-source-picture area 51c is transferred to the forward-source-picture area 51a, overwriting the picture data of the frame F1 stored earlier in the source-picture area 51b. The processed picture data of the frame F4 is stored in the referenced-source-picture area 51b, overwriting the picture data on the frame F2 stored earlier in the referenced-source-picture area 51b. Then, the processed picture data of the frame F5 is stored in the backward-source-picture area 51c, overwriting the picture data of the frame F3 which has been transferred to the forward-source-picture area 51a any way. The operations described above are repeated to process the subsequent frames of the GOP.

[0038] Signals of each picture stored in the frame-memory unit 51 are read out by a prediction-mode switching circuit 52 to undergo a preparatory operation for a frame-prediction mode or a field-prediction mode, that is, the type of processing to be carried out by a processing unit 53.

[0039] Then, in either the frame-prediction mode or the field-prediction mode, the signals are subjected to computing to obtain intra-picture prediction encoding, such as, forward, backward and forward & backward prediction under control executed by an intra-picture-processing/forward/backward/forward & backward prediction determining circuit 54. The type of processing carried out in the processing unit 53 is determined in accordance with a prediction-error signal representing a difference between a referenced picture and a prediction picture for the referenced picture. A referenced picture is a picture subjected to the processing and a prediction picture is a picture preceding or succeeding the referenced picture. For this reason, the motion-vector detecting circuit 50 (strictly speaking, the prediction-mode switching circuit 52 employed in the vector detecting circuit 50 as will be described later) generates a sum of the absolute values of prediction-error signals for use in determination of the type of processing carried out by the processing unit 53. In place of a sum of the absolute values of prediction-error signals, a sum of the squares of prediction-error signals can also be used for such determination.

[0040] The prediction-mode switching circuit 52 carries out the following preparatory operation for processing to be carried out by the processing unit 53 in the frame prediction mode and the field prediction mode.

[0041] The prediction-mode switching circuit 52 receives four luminance blocks [Y1] to [Y4] supplied thereto, by the motion-vector detecting circuit 50. In each block, data of lines of odd fields is mixed with data of lines of even fields as shown in Fig. 8. The data may be passed on to the processing unit 53 as it is. Processing of data with a configuration shown in Fig. 8 to be performed by the processing unit 53 is referred to as processing in the frame-prediction mode wherein prediction processing is carried out for each macroblock comprising the four luminance blocks and a motion vector corresponds to four luminance blocks.

[0042] The prediction-mode switching circuit 52 then reconfigures the signal supplied by the motion-vector detecting circuit 50. In place of the signal with the configuration shown in Fig. 8, the signal with the configuration shown in Fig. 9 may be passed on to the processing unit 53. As shown in Fig. 9, the two luminance blocks [Y1] and [Y2] are each composed of typically only dots of lines of odd fields whereas the other two luminance blocks [Y3] and [Y4] are each composed of typically only dots of lines of even fields. Processing of data with the configuration shown in Fig. 9 to be carried out by the processing unit 53 is referred to as processing in the field-prediction mode wherein a motion vector corresponds to the two luminance blocks [Y1] and [Y2] whereas another motion vector corresponds to the other two luminance blocks [Y3] and [Y4].

[0043] The prediction-mode switching circuit 52 selects the data with the configuration shown in Fig. 8 or Fig. 9 to be supplied to the processing unit 53 as follows. The prediction-mode switching circuit 52 computes a sum of the absolute values of prediction errors computed for the frame-prediction mode, that is, for the data supplied by the motion-vector detecting circuit 50 with the configuration shown in Fig. 8, and a sum of the absolute values of prediction errors computed for the

field-prediction mode, that is, for the data with the configuration shown in Fig. 9 obtained as a result of conversion of the data with the configuration shown in Fig. 8. It should be noted that the prediction errors will be described in detail later. The prediction-mode switching circuit 52 then compares the sum of the absolute values of prediction errors computed for data in order to determine which mode produces the smaller sum. Then, the prediction-mode switching circuit 52 selects either one of the configuration shown in Fig. 8 or Fig. 9 for the frame-prediction mode or the field-prediction mode respectively that produces the smaller sum. The prediction-mode switching circuit 52 finally outputs data with the selected configuration to the processing unit 53 for processing the data in the mode corresponding to the selected configuration.

[0044] It should be noted that, in actuality, the prediction-mode switching circuit 52 is included in the motion-vector detecting circuit 50. That is to say, the preparation of data with the configuration shown in Fig. 9, the computation of the absolute values, the comparison of the absolute values, the selection of the data configuration and the operation to output data with the selected configuration to the processing unit 53 are all carried out by the motion-vector detecting circuit 50 and the prediction-mode switching circuit 52 merely outputs the signal supplied by the motion-vector detecting circuit 50 to the later stage of processing unit 53.

[0045] It should be noted that, in the frame-prediction mode, a color-difference signal is supplied to the processing unit 53 with data of lines of odd fields mixed with data of lines of even fields as shown in Fig. 8. In the field-prediction mode shown in Fig. 9, on the other hand, four lines of the upper half of the color-difference block Cb are used as a color-difference signal of odd fields corresponding to the luminance blocks [Y1] and [Y2] while four lines of the lower half of the color-difference block Cb are used as a color-difference signal of even fields corresponding to the luminance blocks [Y3] and [Y4]. By the same token, four lines of the upper half of the color-difference block Cr are used as a color-difference signal of odd fields corresponding to the luminance blocks [Y1] and [Y2] while four lines of the lower half of the color-difference block Cr are used as a color-difference signal of even fields corresponding to the luminance blocks [Y3] and [Y4].

[0046] As described above, the motion-vector detecting circuit 50 outputs a sum of the absolute values of prediction errors to the prediction determining circuit 54 for use in determination of whether the processing unit 53 should carry out intra-picture prediction, forward prediction, backward prediction or forward & backward prediction

[0047] To put it in detail, a sum of the absolute values of prediction errors in intra-picture prediction is found as follows. For the intra-picture prediction, the motion-vector detecting circuit 50 computes a difference between the absolute value $|\Sigma Aij|$ of a sum Σ Aij of signals Aij of macroblocks of a referenced picture and a sum $\Sigma|Aij|$ of the absolute values $|Aij|$ of the signals Aij of the macroblocks of the same referenced picture. For the forward prediction, the sum of the absolute values of prediction errors is a sum $\Sigma|Aij-Bij|$ of the absolute values $|Aij-Bij|$ of differences (Aij-Bij) between signals Aij of macroblocks of a referenced picture and signals Bij of macroblocks of a forward-prediction picture or a preceding picture. The sum of the absolute values of prediction errors for the backward prediction is found in the same way as that for the forward prediction, except that the prediction picture used in the backward prediction is a backward-prediction picture or a succeeding picture. As for the forward & backward prediction, averages of signals Bij of macroblocks of both the forward-prediction picture and the backward-prediction picture are used in the computation of the sum.

[0048] The sum of the absolute values of prediction errors for each technique of prediction is supplied to the prediction determining circuit 54 which selects the forward prediction, the backward prediction or the forward & backward prediction with a smallest sum as a sum of absolute value of prediction error for an inter-picture prediction. The prediction determining circuit 54 further compares the smallest sum with the sum for the intra-picture prediction and selects either the inter-picture prediction or the intra-picture prediction with the much smaller sum as a prediction mode of processing to be carried out by the processing unit 53. To be more specific, if the sum for the intra-picture prediction is found smaller than the smallest sum for the inter-picture prediction, the prediction determining circuit 54 selects the intra-picture prediction as a type of processing to be carried out by the processing unit 53. If the smallest sum for the inter-picture prediction is found smaller than the sum for the intra-picture prediction, on the other hand, the prediction determining circuit 54 selects the inter-picture prediction as a type of processing to be carried out by the processing unit 53. As described above, the inter-picture prediction represents the forward prediction, the backward prediction or the forward & backward prediction selected as a prediction mode of processing with the smallest sum. The prediction mode is determined for each picture (or frame) while the frame-prediction or field-prediction mode is determined for each group of pictures.

[0049] As described above, the motion-vector detecting circuit 50 outputs signals of macroblocks of a referenced picture for either the frame-prediction mode or the field-prediction mode selected by the prediction-mode switching circuit 52 to the processing unit 53 by way of the prediction-mode switching circuit 52. At the same time, the motion-vector detecting circuit 50 also detects a motion vector between a referenced picture and a prediction picture for one of the four prediction modes selected by the prediction determining circuit 54. The motion-vector detecting circuit 50 then outputs the motion vector to a variable-length-encoding circuit 58

and a motion-compensation circuit 64. In this way, the motion-vector detecting circuit 50 outputs a motion vector that corresponds to the minimum sum of the absolute values of prediction errors for the selected prediction mode as described earlier.

[0050] While the motion-vector detecting circuit 50 is reading out picture data of an I-picture, the first frame of a GOP, from the forward source picture area 51a, the prediction determining circuit 54 sets the intra-picture prediction, strictly speaking, the intra-frame or intra-field prediction, as a prediction mode, setting a switch 53d employed in the processing unit 53 at a contact point a. With the switch 53d set at this position, the data of the I-picture is supplied to a DCT-mode switching circuit 55. As will be described later, the intra-picture prediction mode is a mode in which no motion compensation is carried out.

[0051] The DCT-mode switching circuit 55 receives the data passed on thereto by the switch 53d from the prediction-mode switching circuit 52 in a mixed state or a frame-DCT mode shown Fig. 10. The DCT-mode switching circuit 55 then converts the data into a separated state or a field-DCT mode shown in Fig. 11. In the frame-DCT mode, the data of lines of odd and even fields are mixed in each of the four luminance blocks. In the field-DCT mode, on the other hand, lines of odd fields are put in two of the four luminance blocks while lines of even fields are put in the other two blocks. The data of the I-picture in either the mixed or separated state will be supplied to a DCT circuit 56.

[0052] Before supplying the data to the DCT circuit 56, the DCT-mode switching circuit 55 compares the encoding efficiency of DCT processing of the data with the lines of odd and even fields mixed with each other with the encoding efficiency of DCT processing of the data with the lines of odd and even fields separated from each other to select the data with a higher efficiency. The frame-DCT mode or the field-DCT mode corresponding to the selected data is determined as the DCT mode.

[0053] The encoding efficiencies are compared with each other as follows. In the case of the data with the lines of odd and even fields mixed with each other as shown in Fig. 10, a difference between the signal of a line of an even field and the signal of a line of an odd field vertically adjacent to the even field is computed. Then, the sum or the square of the absolute value of the difference is found. Finally, the sum of the absolute values or the squares of differences between two adjacent even and odd fields is calculated.

[0054] In the case of the data with the lines of odd and even fields separated from each other as shown in Fig. 11, a difference between the signals of lines of vertically adjacent even fields and a difference between the signals of lines of vertically adjacent odd fields are computed. Then, the sum or the square of the absolute value of each difference is found. Finally, the sum of the absolute values or the squares of all differences between two adjacent even fields and two adjacent odd fields is calculated.

[0055] The sum calculated for the data shown in Fig. 10 is compared with the sum calculated for the data shown in Fig. 11 to select a DCT mode. To be more specific, if the former is found smaller than the latter, the frame-DCT mode is selected. If the latter is found smaller than the former, on the other hand, the field-DCT mode is selected.

[0056] Finally, the data with a configuration corresponding to the selected DCT mode is supplied to the DCT circuit 56, and at the same time, a DCT flag used for indicating the selected DCT mode is supplied to the variable-length-encoding circuit 58 and the motion-compensation circuit 64.

[0057] Comparison of the frame-prediction and field-prediction modes of Figs. 8 and 9 determined by the prediction-mode switching circuit 52 with the DCT modes of Figs. 10 and 11 determined by the DCT-mode switching circuit 55 clearly indicates that the data structures of the frame-prediction and field-prediction modes are substantially the same as the data structures of the frame-DCT mode and field-DCT modes respectively with regards to the luminous block.

[0058] If the prediction-mode switching circuit 52 selects the frame-prediction mode in which odd and even lines are mixed with each other, it is quite within the bounds of possibility that the DCT-mode switching circuit 55 also selects the frame-DCT prediction mode with odd and even lines mixed. If the prediction-mode switching circuit 52 selects the field-prediction mode in which odd and even fields are separated from each other, it is quite within the bounds of possibility that the DCT-mode switching circuit 55 also selects the field-DCT mode with the data of odd and even fields separated.

[0059] It should be noted, however, that a selected DCT-mode does not always correspond to a selected prediction mode. In any circumstances, the prediction-mode switching circuit 52 selects a frame-prediction or field-prediction mode that provides a smaller sum of the absolute values of prediction errors and the DCT-mode switching circuit 55 selects a DCT mode giving a better encoding efficiency.

[0060] As described above, the data of the I-picture is output by the DCT-mode switching circuit 55 to the DCT circuit 56. The DCT circuit 56 then carries out DCT processing on the data for conversion into DCT coefficients which are then supplied to a quantization circuit 57. The quantization circuit 57 then carries out a quantization process at a quantization scale adjusted to the amount of data stored in a transmission buffer 59 which is fed back to the quantization circuit 57 as will be described later. The data of the I-picture completing the quantization process is then supplied to a variable-length-encoding circuit 58.

[0061] The variable-length-encoding circuit 58 receives the data of the I-picture supplied from the

quantization circuit 57, converting the picture data into variable-length code such as the Huffman code, at the quantization scale supplied thereto also by the quantization circuit 57. The variable-length code is then stored in the transmission buffer 59.

[0062] In addition to the picture data and the quantization scale supplied by the quantization circuit 57, the variable-length-encoding circuit 58 also receives information on a prediction mode from the prediction determining circuit 54, a motion vector from the motion-vector detecting circuit 50, a prediction flag from the prediction-mode switching circuit 52 and a DCT flag from the DCT-mode switching circuit 55. The information on a prediction mode indicates which type of processing is carried out by the processing unit among intra-picture encoding, forward-prediction encoding, backward-prediction encoding or forward & backward-prediction encoding. The prediction flag indicates that data supplied from the prediction-mode switching circuit 52 to the processing unit 53 is whether in the frame-prediction mode or field-prediction mode. The DCT flag indicates that data supplied from the DCT-mode switching circuit 55 to the DCT circuit 56 is set whether in the frame-DCT or field-DCT mode.

[0063] The transmission buffer 59 temporarily stores input data, feeding the amount of the data stored therein back to the quantization circuit 57. When the amount of data stored in the transmission buffer 59 exceeds an upper limit of an allowable range, the quantization control signal increases the quantization scale of the quantization circuit 57 to reduce the amount of data obtained as a result of the quantization. When the amount of data stored in the transmission buffer 59 becomes smaller than a lower limit of the allowable range, on the other hand, the quantization control signal decreases the quantization scale of the quantization circuit 57 to raise the amount of data obtained as a result of the quantization. In this way, an overflow and an underflow can be prevented from occurring in the transmission buffer 59.

[0064] Then, the data stored in the transmission buffer 59 is read back at predetermined timing to be supplied to the recording circuit 19 for recording the data onto the recording medium 3 which serves as a transmission line.

[0065] The data of the I-picture and the quantization scale output by the quantization circuit 57 are supplied also to an inverse-quantization circuit 60 for carrying out inverse quantization on the data at an inverse-quantization scale corresponding to the quantization scale. Data output by the inverse-quantization circuit 60 is then supplied to an IDCT (Inverse Discrete Cosine Transform) circuit 61 for carrying out inverse discrete cosine transformation. Finally, data output by the IDCT circuit 61 is supplied to a frame-memory unit 63 by way of a processor 62 to be stored in a forward-prediction picture area 63a of the frame-memory unit 63.

[0066] A GOP supplied to the motion-picture detecting circuit 50 to be processed thereby comprises a sequence of pictures I-, B-, P-, B-, P-, B-, etc. In this case, after processing data of the first frame as an I-picture as described above, data of the third frame is processed as a P-picture prior to processing of data of the second frame as a B-picture. This is because a B-picture may be subjected to backward prediction which involves the succeeding P-picture and backward prediction can not be carried out unless the succeeding P-picture has been processed in advance. It should be noted that the data of the I-picture is transferred from the prediction-mode switching circuit 52 to the processing unit 53 in the format for the frame-prediction or field-prediction mode set by the prediction-mode switching circuit 52 for the GOP and always processed by the processing unit 53 in the intra-frame prediction mode as described earlier.

[0067] For the reason described above, after processing data of the first frame as an I-picture, the motion-picture detecting circuit 50 starts processing of the P-picture which has been stored in the backward-source-picture area 51c. Then, the prediction-mode switching circuit 52 computes a sum of the absolute values of differences between frames or prediction errors for data of the P-picture supplied thereto by the motion-vector detecting circuit 50 with the macroblock taken as a unit for each prediction mode and supplies the sum to the prediction determining circuit 54 as described above. The data of the P-picture itself is transferred to the processing unit 53 in the format for the frame-prediction or field-prediction mode which has been set by the prediction-mode switching circuit 52 for the GOP of the P-picture when the I-picture of the first frame of the GOP was input. On the other hand, the prediction determining circuit 54 determines the prediction mode in which the data of the P-picture is to be processed by the processing unit 53, that is, selects either the intra-picture, forward, backward or forward & backward prediction as the type of processing to be carried out by the processing unit 53 on the data of the P-picture on the basis of the sum of the absolute values of prediction errors computed by the prediction-error switching circuit 52 for each prediction mode. Strictly speaking, in the case of a P-picture, the type of processing can be the intra-picture or forward-prediction mode as described above.

[0068] In the first place, in the intra-picture prediction mode, the processing unit 53 sets the switch 53d at the contact point a. Thus, the data of the P-picture is transferred to the transmission line by way of the DCT-mode switching circuit 55, the DCT circuit 56, the quantization circuit 57, the variable-length-encoding circuit 58 and the transmission buffer 59 as is the case with an I-picture. The data of the P-picture is also supplied to the frame-memory unit 63 to be stored in the backward-prediction picture area 63b thereof by way of the quantization circuit 57, the inverse-quantization circuit 60, the IDCT circuit 61 and the processor 62

[0069] In the second place, in the forward-predict on

mode, the processing unit 53 sets the switch 53d at a contact point b and the motion-compensation circuit 64 reads out data from the forward-prediction picture area 63a of the frame-memory unit 63, carrying out motion compensation on the data in accordance with a motion vector supplied to the motion-compensation circuit 64 by the motion-vector detecting circuit 50. In this case, the data stored in the forward-prediction picture area 63a is the data of the I-picture. That is to say, informed of the forward-prediction mode by the prediction determining circuit 54, the motion-compensation circuit 64 generates data of a forward-prediction picture data by reading the data of the I-picture from a read address in the forward-prediction picture area 63a. The read address is a position shifted away from the position of a macroblock currently output by the motion-vector detecting circuit 50 by a distance corresponding to the motion vector.

[0070] The data of the forward-prediction picture read out by the motion-compensation circuit 64 is associated with the data of the referenced picture, that is, the P-picture, and supplied to a processor 53a employed in the processing unit 53. The processor 53a subtracts the data of the forward-prediction picture, that is, the I-picture, supplied by the motion-compensation circuit 64 from the data of macroblocks of the referenced picture supplied by the prediction-mode switching circuit 52 to find a difference or an error in prediction. The difference data is transferred to the transmission line by way of the DCT-mode switching circuit 55, the DCT circuit 56, the quantization circuit 57, the variable-length-encoding circuit 58 and the transmission buffer 59. The difference data is also locally decoded by the inverse-quantization circuit 60 and the IDCT circuit 61 and a result obtained from the local decoding is supplied to the processor 62.

[0071] The data of the forward-prediction picture supplied to the processor 53a by the motion-compensation circuit 64 is also fed to the processor 62. In the processor 62, the data of the forward-prediction picture is added to the difference data output by the IDCT circuit 61 in order to produce the data of the original P-picture. The data of the original P-picture is then stored in the backward-prediction picture area 63b of the frame-memory unit 63.

[0072] After the pieces of data of the I-picture and the P-picture have been stored in the forward-prediction picture area 63a and the backward-prediction picture area 63b of the frame-memory unit 63 respectively as described above, the processing of the second frame of the B-picture is started by the motion-vector detecting circuit 50. The B-picture is processed by the prediction-mode switching circuit 52 in the same way as the P-picture described above except that, in the case of the B-picture, the type of processing determined by the prediction determining circuit 54 can be the backward-prediction mode or the forward & backward-prediction mode in addition to the intra-picture prediction mode or the forward-prediction mode.

[0073] In the case of the intra-picture prediction mode or the forward-prediction mode, the switch 53d is set at the contact point a or b respectively as described above as is the case with the P-picture. In this case, data of the B-picture output by the prediction-mode switching circuit 52 is processed and transferred in the same way as the P-picture described above.

[0074] In the case of the backward-prediction mode or the forward & backward-prediction mode, on the other hand, the switch 53d is set at a contact point c or d respectively.

[0075] In the backward-prediction mode wherein the switch 53d is set at the contact point c, the motion-compensation circuit 64 reads out data from the backward-prediction picture area 63b of the frame-memory unit 63, carrying out motion compensation on the data in accordance with a motion vector supplied to the motion-compensation circuit 64 by the motion-vector detecting circuit 50. In this case, the data stored in the backward-prediction picture area 63b is the data of the P-picture. That is to say, informed of the backward-prediction mode by the prediction determining circuit 54, the motion-compensation circuit 64 generates data of a backward-prediction picture by reading the data of the P-picture from a read address in the backward-prediction picture area 63b. The read address is a position shifted away from the position of a macroblock currently output by the motion-vector detecting circuit 50 by a distance corresponding to the motion vector.

[0076] The data of the backward-prediction picture read out by the motion-compensation circuit 64 is associated with the data of a referenced picture, that is, the B-picture, and supplied to a processor 53b employed in the processing unit 53. The processor 53b subtracts the data of the backward-prediction picture, that is, the P-picture, supplied by the motion-compensation circuit 64 from the data of macroblocks of the referenced picture supplied by the prediction-mode switching circuit 52 to find a difference or an error in prediction. The difference data is transferred to the transmission line by way of the DCT-mode switching circuit 55, the DCT circuit 56, the quantization circuit 57, the variable-length-encoding circuit 58 and the transmission buffer 59.

[0077] In the forward & backward-prediction mode wherein the switch 53d is set at the contact point d, on the other hand, the motion-compensation circuit 64 reads out the data of the I-picture, in this case, from the forward-prediction picture area 63a of the frame-memory unit 63 and the data of the P-picture from the backward-prediction picture area 63b, carrying out motion compensation on the data in accordance with motion vectors supplied to the motion-compensation circuit 64 by the motion-vector detecting circuit 50.

[0078] That is to say, informed of the forward & backward-prediction mode by the prediction determining circuit 54, the motion-compensation circuit 64 generates data of forward & backward-prediction pictures by reading the data of the I- and P-pictures from read

addresses in the forward-prediction picture area 63a and the backward-prediction picture area 63b respectively. The read addresses are positions shifted away from the position of a macroblock currently output by the motion-vector detecting circuit 50 by distances corresponding to the motion vectors. In this case, there are 2 motion vectors, namely, motion vectors for the forward & backward-prediction pictures.

[0079] The data of the forward & backward-prediction pictures read out by the motion-compensation circuit 64 is associated with the data of the referenced picture and supplied to a processor 53c employed in the processing unit 53. The processor 53c subtracts the data of the prediction pictures supplied by the motion-compensation circuit 64 from the data of macroblocks of the referenced picture supplied by the prediction-mode switching circuit 52 employed in the motion-vector detecting circuit 50 to find a difference or an error in prediction. The difference data is transferred to the transmission line by way of the DCT-mode switching circuit 55, the DCT circuit 56, the quantization circuit 57, the variable-length-encoding circuit 58 and the transmission buffer 59.

[0080] Never used as a prediction picture for another frame, a B-picture is not stored in the frame-memory unit 63.

[0081] It should be noted that, typically, the forward-prediction picture area 63a and the backward-prediction picture area 63b of the frame-memory unit 63 are implemented as memory banks which can be switched from one to another. Thus, in an operation to read a forward-prediction picture, the frame-memory unit 63 is set to the forward-prediction picture area 63a. In an operation to read a backward-prediction picture, on the other hand, the frame-memory unit 63 is set to the backward-prediction picture area 63b.

[0082] While the above description is focused on luminance blocks, the color-difference blocks are also processed and transferred in macroblock units shown in Figs. 8 to 11 in the same way as the luminance blocks. It should be noted that, as a motion vector in the processing of the color-difference blocks, components of the motion vector of the luminance blocks associated with the color-difference blocks in the vertical and horizontal directions are used with their magnitudes each cut in half.

[0083] Fig. 12 is a block diagram showing the configuration of the decoder 31 employed in the moving-picture encoding/decoding apparatus shown in Fig. 5. Encoded picture data transmitted through the transmission line implemented by the recording medium 3 is received by the decoder 31, by way of the playback circuit 30 of the moving-picture encoding/decoding apparatus and then stored temporarily in a reception buffer 81 employed in the decoder 31. Then, the picture data is supplied to a variable-length-decoding circuit 82 employed in a decoding circuit 90 of the decoder 31. The variable-length-decoding circuit 82 carries out variable-length decoding on the picture data read out from the reception buffer 81, outputting a motion vector, information on the prediction mode, a frame/field-prediction flag and a frame/field DCT-flag to a motion compensation-circuit 87 and a quantization scale as well as decoded picture data to an inverse-quantization circuit 83.

[0084] The inverse-quantization circuit 83 carries out inverse quantization on the picture data supplied thereto by the variable-length-decoding circuit 82 at the quantization scale also received from the variable-length-decoding circuit 82. The inverse-quantization circuit 83 outputs DCT coefficients data obtained as a result of the inverse quantization to an IDCT circuit 84 for carrying out IDCT (inverse discrete cosine transformation), supplying the result of the IDCT to a processor 85.

[0085] In the case of an I-picture, the picture data supplied to the processor 85 by the IDCT circuit 84 is output as it is, by the processor 85 to a frame-memory unit 86 to be stored in a forward-prediction picture area 86a of the frame-memory unit 86. The data of the I-picture stored in the forward-prediction picture area 86a will be used for generation of data of a forward-prediction picture for picture data of a P- or B-picture supplied to the processor 85 after the I-picture in the forward-prediction mode. The data of the I-picture is also output to a format converting circuit 32 employed in the moving-picture encoding/decoding apparatus shown in Fig. 5.

[0086] When the picture data supplied by the IDCT circuit 84 is the data of P-picture having the picture data leading ahead by one frame, that is, the picture data of the I-picture, is read out by the motion-compensation circuit 87 from the forward-prediction picture area 86a of the frame-memory unit 86. In the motion-compensation circuit 87, the picture data of the I-picture undergoes motion compensation for a motion vector supplied by the variable-length-decoding circuit 82. The picture data completing the motion compensation is then supplied to the processor 85 to be added to the picture data supplied by the IDCT circuit 84 which is actually difference data. The result of the addition, that is, data of the decoded P-picture is supplied to the frame-memory unit 86 to be stored in the backward-prediction picture area 86b of the frame-memory unit 86 as described above. The data of the P-picture stored in the backward-prediction picture area 86b will be used for generation of data of a backward-prediction picture for picture data of a B-picture supplied to the processor 85 thereafter in the backward-prediction mode.

[0087] On the other hand, picture data of a P-picture processed by the signal encoding apparatus 1 in the intra-frame prediction mode is output by the processor 85 as it is without undergoing any processing, to the backward-prediction picture area 86b as is the case with an I-picture.

[0088] Since the P-picture will be displayed after a B-picture to be processed after the P-picture at this point

of time, the P-picture is not output to the format converting circuit 32. Much like the encoder 18, the decoder 31 processes and transfers the P-picture prior to the B-picture even though the P-picture is received after the B-picture.

[0089] Picture data of a B-picture output by the IDCT circuit 84 is processed by the processor 85 in accordance with the information on the prediction mode supplied by the variable-length-decoding circuit 82. To be more specific, the processor 85 may output the picture data as it is in the intra-picture-prediction mode as is the case with the I-picture or process the picture data in the forward-prediction, backward-prediction or forward & backward-prediction mode. In the forward-prediction, backward-prediction or forward & backward-prediction mode, the motion-compensation circuit 87 reads out the data of the I-picture stored in the forward-prediction picture area 86a, the data of the P-picture stored in the backward-prediction picture area 86b or the data of the I and P-pictures stored in the forward-prediction picture area 86a and the backward-prediction picture area 86b of the frame-memory unit 86 respectively. The motion-compensation circuit 87 then carries out motion compensation on the picture data read out from the frame-memory unit 86 in accordance with a motion vector output by the variable-length-decoding circuit 82 to produce a prediction picture. In the case of the intra-picture-prediction mode described above, no prediction picture is generated because no prediction picture is required by the processor 85.

[0090] The prediction picture experiencing the motion compensation in the motion-compensation circuit 87 is added by the processor 85 to the picture data of the B-picture, strictly speaking difference data, output by the IDCT circuit 84. Data output by the processor 85 is then supplied to the format converting circuit 32 as is the case with the I-picture.

[0091] Since the data output by the processor 85 is the picture data of a B-picture, however, the data is not required in generation of a prediction picture. Thus, the data output by the processor 85 is not stored in the frame-memory unit 86.

[0092] After the data of the B-picture has been output, the data of the P-picture is read out by the motion-compensation circuit 87 from the backward-prediction picture area 86b and supplied to the processor 85. This time, however, the data is not subjected to motion compensation since the data has experienced the motion compensation before being stored in the backward-prediction picture area 86b.

[0093] The decoder 31 does not include counterpart circuits of the prediction-mode switching circuit 52 and the DCT-mode switching circuit 55 employed in the encoder 18 shown in Fig. 5 because the counterpart processing, that is, processing to convert the signal formats with even and odd fields separated from each other as shown in Figs. 9 and 11 back into the signal formats with even and odd fields mixed with each other as shown in Figs. 8 and 10 respectively is carried out by the motion-compensation circuit 87.

[0094] While the above description is focused on luminance signals, the color-difference signals are also processed and transferred in macroblock units shown in Figs. 8 to 11 in the same way as the luminance signals. It should be noted that, as a motion vector in the processing of the color-difference signals, components of the motion vector of the luminance signals associated with the color-difference blocks in the vertical and horizontal directions are used with their magnitudes each cut in half.

[0095] Fig. 13 is a diagram showing the quality of encoded pictures in terms of an SNR (Signal-to-Noise Ratio). As shown in the figure, the quality of a picture much depends on the type of the picture. To be more specific, transmitted I- and P-pictures have high qualities, but B-picture has poorer qualities respectively. Deliberate variations in picture quality shown in Fig. 13 is a sort of technique for making use of the human being' s characteristic of the sense of sight. That is to say, by varying the quality, the quality appears better to the sense of sight than a case in which an average of all picture qualities is used. Control to vary the picture quality is executed by the quantization circuit 57 employed in the encoder 18 shown in Fig. 7.

[0096] Figs. 14 and 15 are diagrams showing the configuration of the transcoder 101.

[0097] Fig. 15 shows the configuration shown in Fig. 14 in more detail. The transcoder 101 converts the GOP structure and the bit rate of an encoded video bitstream supplied to the video decoding apparatus 102 into a GOP structure and a bit rate desired by the operator or specified by the host computer, respectively. The function of the transcoder 101 is explained by assuming that, in actuality, three other transcoders each having all but the same function as the transcoder 101 are connected at the front stage of the transcoder 101. In order to convert the GOP structure and the bit rate of bitstream into one of a variety of GOP structures and one of a variety of bit rates respectively, transcoders of the first, second and third generations are connected in series and the transcoder 101 of the fourth generation shown in Fig. 15 is connected behind the series connection of the transcoders of the first, second and third generation. It should be noted that the transcoders of the first, second and third generations themselves are not shown in Fig. 15.

[0098] In the following description, an encoding process carried out by the transcoder of the first generation is referred to as an encoding process of the first generation and an encoding process carried out by the transcoder of the second generation connected after the transcoder of the first generation is referred to as an encoding process of the second generation. Likewise, an encoding process carried out by the transcoder of the third generation connected after the transcoder of the second generation is referred to as an encoding

process of the third generation and an encoding process carried out by the fourth transcoder connected after the transcoder of the third generation, that is, the transcoder 101 shown in Fig. 15, is referred to as an encoding process of the fourth generation. In addition, encoding parameters used as well as obtained as a result of the encoding process of the first generation are referred to as encoding parameters of the first generation and encoding parameters used as well as obtained as a result of the encoding process of the second generation are referred to as encoding parameters of the second generation. Similarly, encoding parameters used as well as obtained as a result of the encoding process of the third generation are referred to as encoding parameters of the third generation and encoding parameters used as well as obtained as a result of the encoding process of the fourth generation are referred to as encoding parameters of the fourth generation or the current encoding parameters.

[0099] First, an encoded video bitstream ST(3rd) of the third generation generated and supplied by the transcoder of the third generation to the transcoder 101 of the fourth generation shown in Fig. 15 is explained. The encoded video bitstream ST(3rd) of the third generation is an encoded video bitstream obtained as a result of the encoding process of the third generation carried out by the third transcoder provided at a stage preceding the transcoder 101 of the fourth generation. In the encoded video bitstream ST(3rd) of the third generation resulting from the encoding process of the third generation, the encoding parameters of the third generation generated in the encoding process of the third generation are described as a sequence_header() function, a sequence_extension() function, a group_of_pictures_header() function, a picture_header() function, a picture_coding_extension() function, a picture_data() function, a slice() function and a macroblock() function on a sequence layer, a GOP layer, a picture layer, a slice layer and a macroblock layer of the encoded video bitstream ST of the third generation, respectively. The fact that the encoding parameters of the third generation used in the encoding process of the third generation are described in the encoded video bitstream ST of the third generation resulting from the encoding process of the third generation conforms to MPEG2 standard and does not reveal any novelty whatsoever.

[0100] A point unique to the transcoder 101 is not the fact that the encoding parameters of the third generation are described in the encoded video bitstream ST of the third generation, but the fact that the encoding parameters of the first and second generations obtained as results of the encoding processes of the first and second generations respectively are included in the encoded video bitstream ST of the third generation. The encoding parameters of the first and second generations are described as history_stream() in a user_data area of the picture layer of the encoded video bitstream ST of the third generation. In the present invention, the history stream described in a user_data area of the picture layer of the encoded video bitstream ST of the third generation is referred to as "history information" and the parameters described as a history stream are referred to as "history parameters". In another way of naming parameters, the encoding parameters of the third generation described in the encoded video bitstream ST of the third generation can also be referred to as current encoding parameters. In this case, the encoding parameters of the first and second generations described as history_stream() in the user-data area of the picture layer of the encoded video bitstream ST of the third generation are referred to as "past encoding parameters" since the encoding processes of the first and second generations are each a process carried out in the past if viewed from the encoding process of the third generation.

[0101] The reason why the encoding parameters of the first and second generations obtained as results of the encoding processes of the first and second generations respectively are also described in the encoded video bitstream ST(3rd) of the third generation in addition to the encoding parameters of the third generation as described above is to avoid deterioration of the picture quality, even if the GOP structure and the bit rate of encoding stream are changed repeatedly in transcoding processes. For example, a picture may be encoded as a P-picture in the encoding process of the first generation and, in order to change the GOP structure of the encoded, video bitstream of the first generation, the picture is encoded as a B-picture in the encoding process of the second generation. In order to further change the GOP structure of the encoded video bitstream of the second generation, the picture is again encoded as a P-picture in the encoding process of the third generation. Since conventional encoding and decoding processes based on the MPEG standard are not 100% reverse processed, the picture quality deteriorates each time encoding and decoding processes are carried out as is generally known. In such a case, encoding parameters such as the quantization scale, the motion vector and the prediction mode are not merely re-computed in the encoding process of the third generation. Instead, the encoding parameters such as the quantization scale, the motion vector and the prediction mode generated in the encoding process of the first generation are re-utilized. The encoding parameters such as the quantization scale, the motion vector and the prediction mode newly generated in the encoding process of the first generation obviously each have a precision higher than the counterpart encoding parameters newly generated in the encoding process of the third generation. Thus, by re-utilizing the encoding parameters generated in the encoding process of the first generation, it is possible to lower the degree to which the picture quality deteriorates even if the encoding and decoding processes are carried out repeatedly.

[0102] The processing described above is exemplified by explaining the decoding and encoding processes carried out by the transcoder 101 of the fourth generation shown in Fig. 15 in detail. The video decoding apparatus 102 decodes an encoded video signal included in the encoded video bitstream ST(3rd) of the third generation by using the encoding parameters of the third generation to generate decoded base-band digital video data. In addition, the video decoding apparatus 102 also decodes the encoding parameters of the first and second generations described as a history stream in the user-data area of the picture layer of the encoded video bitstream ST(3rd) of the third generation. The configuration and the operation of the video decoding apparatus 102 are described in detail by referring to Fig. 16 as follows.

[0103] Figs. 16 is a diagram showing a detailed configuration of the video decoding apparatus 102. As shown in the figure, the video decoding apparatus 102 comprises a buffer 81 for buffering a supplied encoded bitstream, a variable-length decoding circuit 112 for carrying out a variable-length decoding process on the encoded bitstream, an inverse-quantization circuit 83 for carrying out inverse quantization on data completing the variable-length decoding process in accordance with a quantization scale supplied by the variable-length decoding circuit 112, an IDCT circuit 84 for carrying out inverse discrete cosine transformation on DCT coefficients completing the inverse quantization, a processor 85 for carrying out motion compensation processing, a frame-memory unit 86 and a motion-compensation circuit 87.

[0104] In order to decode the encoded video bitstream ST(3rd) of the third generation, the variable-length decoding circuit 112 extracts the encoding parameters of the third generation described on the picture layer, the slice layer and the macroblock layers of the encoded video bitstream ST(3rd) of the third generation. Typically, the encoding parameters of the third generation extracted by the variable-length decoding circuit 112 include picture_coding_type representing the type of the picture, quantiser_scale_code representing the size of the quantization-scale step, macrobloc_type representing the prediction mode, motion_vector representing the motion vector, frame/field_motion_type indicating a frame prediction mode or a field prediction mode and dct_type indicating a frame-DCT mode or a field-DCT mode. The quantiser_scale_code encoding parameter is supplied to the inverse-quantization circuit 83. On the other hand, the rest of the encoding parameters such as picture_coding_type, quantiser_scale_code, macroblock_type, motion_vector, frame/field_motion_type and dct_type are supplied to the motion-compensation circuit 87.

[0105] The variable-length decoding circuit 112 extracts not only the above mentioned encoding parameters required for decoding the encoded video bitstream ST(3rd) of the third generation, but all other encoding parameters of the third generation to be transmitted to a transcoder of the fifth generation connected behind the transcoder 101 shown in Fig. 15 as history information of the third generation from the sequence layer, the GOP layer, the picture layer, the slice layer and the macroblock layer of the encoded video bitstream ST(3rd) of the third generation. It is needless to say that the above encoding parameters of the third generation such as picture_coding_type, quantiser_scale_code, macrobloc_type, motion_vector, frame/field_motion_type and dct_type used in the process of the third generation as described above are also included in the history information of the third generation. The operator or the host computer determines in advance what encoding parameters are to be extracted as history information in accordance with the transmission capacity.

[0106] In addition, the variable-length decoding circuit 112 also extracts user data described in the user-data area of the picture layer of the encoded video bitstream ST(3rd) of the third generation, supplying the user data to the history decoding apparatus 104.

[0107] The history decoding apparatus 104 extracts the encoding parameters of the first and second generations described as history information from the user data extracted from the picture layer of the encoded video bitstream ST(3rd) of the third generation. To put it concretely, by analyzing the syntax of the user data received from the variable-length decoding circuit 112, the history decoding apparatus 104 is capable of detecting unique History_Data_Id described in the user data and using it for extracting converted_history_stream(). Then, by fetching 1-bit marker bits (marker_bit) inserted into converted_history_stream() at predetermined intervals, the history decoding apparatus 104 is capable of acquiring history_stream(). By analyzing the syntax of history_stream(), the history decoding apparatus 104 is capable of extracting the encoding parameters of the first and second generations recorded in history_stream(). The configuration and the operation of the history decoding apparatus 104 will be described in detail later.

[0108] In order to eventually supply the encoding parameters of the first, second and third generations to the video encoding apparatus 106 for carrying out the encoding process of the fourth generation, the history information multiplexing apparatus 103 multiplexes the encoding parameters of the first, second and third generations in base-band video data decoded by the video decoding apparatus 102. The history information multiplexing apparatus 103 receives the base-band video data from the video decoding apparatus 102, the encoding parameters of the third generation from the variable-length decoding circuit 112 employed in the video decoding apparatus 102 and the encoding parameters of the first as well as second generations from the history decoding apparatus 104, multiplexing the encoding parameters of the first, second and third generations in

base-band video data. The base-band video data with the encoding parameters of the first, second and third generations multiplexed therein is then supplied to the history information separating apparatus 105.

[0109] Next, a technique of multiplexing the encoding parameters of the first, second and third generations in the base-band video data is explained by referring to Figs. 17 and 18. Fig. 17 is a diagram showing a macroblock composed of a luminance-signal portion and a color-difference-signal portion with the portions each comprising 16 pixels 16 pixels as defined in accordance with MPEG standard. One of the portions comprising 16 pixels 16 pixels is composed of 4 sub-blocks Y[0], Y[1], Y[2] and Y[3] for the luminance signal whereas the other portion is composed of 4 sub-blocks Cr[0], Cr[1], Cb[0] and Cb[1] for the color-difference signal. The sub-blocks Y[0], Y[1], Y[2] and Y[3] and the sub-blocks Cr[0], Cr[1], Cb[0] and Cb[1] each comprise 8 pixels 8 pixels.

[0110] Fig. 18 is a diagram showing a format of video data. Defined in accordance with an RDT601 recommended by ITU, the format represents the so-called D1 format used in the broadcasting industry. Since the D1 format is standardized as a format for transmitting video data, 1 pixel of video data is expressed by 10 bits.

[0111] Base-band video data decoded in conformity with the MPEG standard is 8 bits in length. In the transcoder provided by the present invention, base-band video data decoded in conformity with the MPEG standard is transmitted by using the 8 high-order bits D9 to D2 of the 10-bit D1 format as shown in Fig. 18. Therefore, the 8-bit decoded video data leaves 2 low-order bits D1 and D2 unallocated in the D1 format. The transcoder provided by the present invention utilizes an unallocated area comprising these unallocated bits for transmitting history information.

[0112] The data block shown in Fig. 18 is a data block for transmitting a pixel in the 8 sub-blocks of a macroblock. Since each sub-block actually comprises 64 (= 8 8) pixels as described above, 64 data blocks each shown in Fig. 18 are required to transmit a volume of data of macroblock comprising the 8 sub-blocks. As described above, the macroblock for the luminance and color-difference signals comprises 8 sub-blocks each composed of 64 (= 8 8) pixels. Thus, the macroblock for the luminance and color-difference signals comprises 8 64 pixels = 512 pixels. Since each pixel leaves 2 bits unallocated as described above, the macroblock for the luminance and color-difference signals has 512 pixels 2 unallocated bits/pixel = 1,024 unallocated bits. By the way, history information of one generation is 256 bits in length. Thus, history information of four (= 1,024/256) previous generations can be superposed on the macroblock of video data for the luminance and color-difference signals. In the example shown in Fig. 18, history information of the first, second and third generations is superposed on the one macroblock of video data by utilizing the 2 low-order bits of D1 and D0.

[0113] The history information separating apparatus 105 extracts base-band video data from the 8 high-order bits of data transmitted thereto as the D1 format and history information from the 2 low-order bits. In the example shown in Fig. 15, the history information separating apparatus 105 extracts base-band video data from transmitted data, supplying the base-band video data to the video encoding apparatus 106. At the same time, the history information separating apparatus 105 extracts history information comprising encoding parameters of the first, second and third generations from the transmitted data, supplying the history information to the signal encoding circuit 106 and the history encoding apparatus 107.

[0114] The video encoding apparatus 106 encodes the base-band video data supplied thereto by the history information separating apparatus 105 into a bitstream having a GOP structure and a bit rate specified by the operator or the host computer. It should be noted that changing the GOP structure means changing the number of pictures included in the GOP, changing the number of P-pictures existing between two consecutive I-pictures or changing the number of B-pictures existing between two consecutive I-pictures or between an I-picture and a P-picture.

[0115] In the embodiment shown in Fig. 15, the supplied base-band video data includes history information of the encoding parameters of the first, second and third generations superposed thereon. Thus, the video encoding apparatus 106 is capable of carrying out an encoding process of the fourth generation by selective reutilization of these pieces of history information so as to lower the degree to which the picture quality deteriorates.

[0116] Fig. 19 is a concrete diagram showing the configuration of an encoder 121 employed in the video encoding apparatus 106. As shown in the figure, the encoder 121 comprises a motion-vector detecting circuit 50, a prediction mode switch circuit 52, a processor 53, a DCT switch circuit 55, a DCT circuit 56, a quantization circuit 57, a variable-length coding circuit 58, a transmission buffer 59, an inverse-quantization circuit 60, an inverse-DCT circuit 61, a processor 62, a frame memory unit 63 and a motion compensation 64. The functions of these circuits are all but the same as those employed in the encoder 18 shown in Fig. 7, making it unnecessary to repeat their explanation. The following description thus focuses on differences between the encoder 121 and the encoder 18 shown in Fig. 7.

[0117] The encoder 121 also includes a controller 70 for controlling the operations and the functions of the other aforementioned components composing the encoder 121. The controller 70 receives an instruction specifying a GOP structure from the operator or the host computer, determining the types of pictures constituting the GOP structure. In addition, the controller 70 also receives information on a target bit rate from the operator or the host computer, controlling the quantization circuit 57 so as to set the bit rate output by the encoder

121 at the specified target bit rate.

[0118] Furthermore, the controller 70 also receives history information of a plurality of generations output by the history information separating apparatus 105, encoding a referenced picture by reutilization of this history information. The functions of the controller 70 are described in detail as follows.

[0119] First, the controller 70 forms a judgment as to whether or not the type of a referenced picture determined from the GOP structure specified by the operator or the host computer matches with the picture type included in the history information. That is to say, the controller 70 forms a judgment as to whether or not the referenced picture has been encoded in the past in the same picture type as the specified picture type.

[0120] The formation of the judgment described above can be exemplified by using the example shown in Fig. 15. The controller 70 forms a judgment as to whether or not the picture type assigned to the referenced picture in the encoding process of the fourth generation is the same type as that of the referenced picture in the encoding process of the first generation, the type of the referenced picture in the encoding process of the second generation or the type of the referenced picture in the encoding process of the third generation.

[0121] If the result of the judgment indicates that the picture type assigned to the referenced picture in the encoding process of the fourth generation is not the same type as that of the referenced picture in an encoding process of any previous generation, the controller 70 carries out a normal encoding process. This result of the judgment implies that this referenced picture was never subjected to the encoding processes of the first, second and third generations previously in the picture type assigned to the referenced picture in the encoding process of the fourth generation. If the result of the judgment indicates that the picture type assigned to the referenced picture in the encoding process of the fourth generation is the same type as that of the referenced picture in an encoding process of a previous generation, on the other hand, the controller 70 carries out a parameter-reuse-encoding-process by reutilization of parameters of the previous generation. This result of the judgment implies that this referenced picture was subjected to the encoding process of the first, second or third generation previously in the picture type assigned to the referenced picture in the encoding process of the fourth generation.

[0122] First, the normal encoding process carried out by the controller 70 is explained. In order to allow the controller 70 to make a decision as to which one of the frame-prediction mode or the field-prediction mode should be selected, the motion-vector detecting circuit 50 detects a prediction error in the frame prediction mode and a prediction error in the field prediction mode, supplying the values of the prediction errors to the controller 70. The controller 70 compares the values with each other, selecting the prediction mode with the smaller prediction error. The prediction mode switch circuit 52 then carries out signal processing to correspond to the prediction mode selected by the controller 70, supplying a signal obtained as a result of the processing to the processing unit 53. With the frame-prediction mode selected, the prediction mode switch circuit 52 carries out signal processing so as to supply the luminance signal to the processing unit 53 as it is on receiving the signal, and carries out signal processing of the color-difference signal so as to mix odd field lines with even field lines as being described earlier by referring to Fig. 8. With the field-prediction mode selected, on the other hand, the prediction mode switch circuit 52 carries out signal processing of the luminance signal so as to make the luminance sub-blocks Y[1] and Y[2] comprise odd field lines and luminance sub-blocks Y[3] and Y[4] comprise even field lines, and carries out signal processing of the color-difference signal so as to make the four upper lines comprise odd field lines and the four lower lines comprise even field lines as described earlier by referring to Fig. 9.

[0123] In addition, in order to allow the controller 70 to make a decision as to which one of the intra-picture prediction mode, the forward prediction mode, the backward prediction mode or the forward & backward prediction mode to be selected, the motion-vector detecting circuit 50 generates a prediction error for each of the prediction modes, supplying the prediction errors to the controller 70. The controller 70 selects a mode having the smallest prediction error as an inter-picture prediction mode from the forward prediction mode, the backward prediction mode or the forward & backward prediction mode. Then, the controller 70 compares the smallest prediction error of the selected inter-picture prediction mode with the prediction error of the intra-picture prediction mode, selecting either the selected inter-picture prediction or the intra-picture prediction mode with a smaller prediction error as a prediction mode. To put it in detail, if the prediction error of the intra-picture prediction mode is found smaller, the intra-picture prediction mode is established. If the prediction error of the inter-picture prediction mode is found smaller, on the other hand, the selected forward prediction, backward prediction mode or forward & backward prediction mode with the smallest prediction error is established. The controller 70 then controls the processor 53 and the motion compensation 64 to operate in the established prediction mode.

[0124] Furthermore, in order to allow the controller 70 to make a decision as to which one of the frame-DCT mode or the field-DCT mode to be selected, the DCT mode switch circuit 55 converts data of the four luminance sub-blocks into a signal with a format of the frame-DCT mode comprising mixed odd and even field lines and a signal with a format of the field-DCT mode comprising separated odd and even field lines, supplying the signals resulting from the conversion to the DCT circuit 56. The DCT circuit 56 computes an encoding

efficiency of DCT processing of the signal comprising mixed odd and even field lines and an encoding efficiency of DCT processing of the signal comprising separated odd and even field lines, supplying the computed encoding efficiencies to the controller 70. The controller 70 compares the encoding efficiencies with each other, selecting a DCT mode with a higher efficiency. The controller 70 then controls the DCT mode switch circuit 55 to work in the selected DCT mode.

[0125] The controller 70 also receives a target bit rate representing a desired bit rate specified by the operator or the host computer and a signal representing the volume of data stored in the transmission buffer 59 or the size of a residual free area left in the buffer 59, generating feedback_q_scale_code for controlling the size of the quantization step used by the quantization circuit 57 on the basis of the target bit rate and the size of the residual free area left in the buffer 59. The feedback_q_scale_code is a control signal generated in accordance with the size of the residual free area left in the transmission buffer 59 so as to prevent an overflow or an underflow from occurring in the buffer 59 and to cause a bitstream to be output from the transmission buffer 59 at the target bit rate. To put it concretely, if the volume of data buffered in the transmission buffer 59 becomes small, for example, the size of the quantization step is reduced so that the number of bits of a picture to be encoded next increases. If the volume of data buffered in the transmission buffer 59 becomes large, on the other hand, the size of the quantization step is increased so that the number of bits of a picture to be encoded next decreases. It should be noted that the size of the quantization step is proportional to feedback_q_scale_code. That is to say, when feddback_q_scale_code is increased, the size of the quantization step rises. When feddback_q_scale_code is reduced, on the other hand, the size of the quantization step decreases.

[0126] Next, the parameter-reuse-encoding process reutilizing encoding parameters which characterizes the transcoder 101 is explained. In order to make the explanation easy to understand, it is assumed that the referenced picture was encoded as an I-picture in the encoding process of the first generation, as a P-picture in the encoding process of the second generation and as a B-picture in the encoding process of the third generation, and has to be encoded again as an I-picture in the current encoding process of the fourth generation. In this case, since the referenced picture was previously encoded in the encoding process of the first generation in the required picture type as the I-picture assigned to the encoding process of the fourth generation, the controller 70 carries out an encoding process by using the encoding parameters of the first generation instead of creating new encoding parameters from the supplied video data. Representatives of such encoding parameters to be reutilized in the encoding process of the fourth generation include quantiser_scale_code representing the size of the quantization-scale step, macrobloc_type representing the prediction mode, motion_vector representing the motion vector, frame/field_motion_type indicating a frame prediction mode or a field prediction mode and dct_type indicating a frame-DCT mode or a field-DCT mode. The controller 70 does not reutilize all encoding parameters received as information history. Instead, the controller 70 reutilizes only encoding parameters judged to be appropriate for reutilization and newly creates encoding parameters for which previous encoding parameters are not suitable for reutilization.

[0127] Next, the parametre-reuse-encoding process reutilizing encoding parameters is explained by focusing on differences from the normal encoding process described earlier. In the normal encoding process, the motion-vector detecting circuit 50 detects a motion vector of a referenced picture. In the parameter-reuse-encoding process reutilizing encoding parameters, on the other hand, the motion-vector detecting circuit 50 does not detect a motion vector of a referenced picture. Instead, the motion-vector detecting circuit 50 reutilizes motion_vector transferred as history information of the first generation. The reason motion_vector of the first generation is used is explained as follows. Since the base-band video data obtained as a result of a decoding process of the encoded bitstream of the third generation has experienced at least three encoding processes, the picture quality thereof is obviously poor in comparison with the original video data. A motion vector detected from video data with a poor picture quality is not accurate. To be more specific, a motion vector supplied to the transcoder 101 of the fourth generation as history information of the first generation is certainly has a precision better than a motion vector detected in the encoding process of the fourth generation. By reutilization of the motion vector received as an encoding parameter of the first generation, the picture quality does not deteriorate during the encoding process of the fourth generation. The controller 70 supplies the motion vector received as the history information of the first generation to the motion compensation 64 and the variable-length coding circuit 58 to be used as the motion vector of a referenced picture being encoded in the encoding process of the fourth generation.

[0128] In the normal processing, the motion-vector detecting circuit 50 detects a prediction error in the frame-prediction mode and a prediction error in the field-prediction mode in order to select either the frame-prediction mode or the field-prediction mode. In the parameter-reuse-encoding processing based on reutilization of encoding parameters, on the other hand, the motion-vector detecting circuit 50 detects neither prediction error in the frame-prediction mode and nor prediction error in the field-prediction mode. Instead, frame/field_motion_type received as history information of the first generation to indicate either the frame-prediction mode or the field-prediction mode is reutilized

This is because the prediction error of each prediction mode detected in the encoding process of the first generation has a precision higher than the prediction error of each prediction mode detected in the encoding process of the fourth generation. Thus, a prediction mode selected on the basis of prediction errors each having a high precision will allow a more optimum encoding process to be carried out. To put it concretely, the controller 70 supplies a control signal representing frame/field_motion_type received as history information of the first generation to the prediction mode switch circuit 52. The control signal drives the prediction mode switch circuit 52 to carry out signal processing according to reutilized frame/field_motion_type.

[0129] In the normal processing, the motion-vector detecting circuit 50 also detects a prediction error in each of the intra-picture prediction mode, the forward prediction mode, the backward prediction mode and the forward & backward prediction mode in order to select one of these prediction modes. In the processing based on reutilization of encoding parameters, on the other hand, the motion-vector detecting circuit 50 detects no prediction errors of these prediction modes. Instead, one of the intra-picture prediction mode, the forward prediction mode, the backward prediction mode and the forward & backward prediction mode indicated by macroblock_type received as history information of the first generation is selected. This is because the prediction error of each prediction mode detected in the process of the first generation has a precision higher than the prediction error of each prediction mode detected in the process of the fourth generation. Thus, a prediction mode selected on the basis of prediction errors each having a high precision will allow a more efficient encoding process to be carried out. To put it concretely, the controller 70 selects a prediction mode indicated by macroblock_type included in the history information of the first generation and controls the processor 53 and the motion-compensation 64 to operate in the selected prediction mode.

[0130] In the normal encoding process, the DCT mode switch circuit 55 supplies both of a signal converted into a format of the frame-DCT mode and a signal converted into a format of the field-DCT mode to the DCT circuit 56 for use in comparison of an encoding efficiency in the frame-DCT mode with an encoding efficiency in the field-DCT mode. In the processing based on reutilization of encoding parameters, on the other hand, neither the signal converted into a format of the frame-DCT mode nor the signal converted into a format of the field-DCT mode is generated. Instead, only processing in a DCT mode indicated by dct_type included in the history information of the first generation is carried out. To put it concretely, the controller 70 reutilizes dct_type included in the history information of the first generation and controls the DCT-mode switching circuit 55 to carry out signal processing in accordance with a DCT mode indicated by dct_type.

[0131] In the normal encoding process, the controller 70 controls the size of the quantization step used in the quantization circuit 57 on the basis of a target bit rate specified by the operator or the host computer and the size of a residual free area left in the transmission buffer 59. In the processing based on reutilization of encoding parameters, on the other hand, the controller 70 controls the size of the quantization step used in the quantization circuit 57 on the basis of a target bit rate specified by the operator or the host computer, the size of a residual free area left in the transmission buffer 59 and a past quantization scale included in history information. It should be noted that, in the following description, the past quantization scale included in history information is referred to as history_q_scale_code. In a history stream to be described later, the quantization scale is referred to as quantiser_scale_code.

[0132] First, the controller 70 generates feedback_q_scale_code representing the current quantization scale as is the case with the normal encoding process. Feedback_q_scale_code is set at such a value determined from the size of the residual free area left in the transmission buffer 59 that neither overflow nor underflow occurs in the transmission buffer 59. Then, history_q_scale_code representing a previous quantization scale included in the history stream of the first generation is compared with feedback_q_scale_code representing the current quantization scale to determine which quantization scale is greater. It should be noted that a large quantization scale implies a large quantization step. If feedback_q_scale_code representing the current quantization scale is found greater than history_q_scale_code representing the largest among a plurality of previous quantization scales, the controller 70 supplies feedback_q_scale_code representing the current quantization scale to the quantization circuit 57. If history_q_scale_code representing the largest previous quantization scale is found greater than feedback_q_scale_code representing the current quantization scale, on the other hand, the controller 70 supplies history_q_scale_code representing the largest previous quantization scale to the quantization circuit 57. The controller 70 selects the largest among a plurality of previous quantization scales included in the history information and the current quantization scale derived from the size of the residual free area left in the transmission buffer 59. In other words, the controller 70 controls the quantization circuit 57 to carry out quantization using the largest quantization step among the quantization steps used in the current encoding process (or the encoding process of the fourth generation) and the previous encoding processes (or the encoding processes of the first, second and third generations). The reason is described as follows.

[0133] Assume that the bit rate of a stream generated in the encoding process of the third generation is 4 Mbps and the target bit rate set for the encoder 121 carrying out the encoding process of the fourth generation

is 15 Mbps. Such a target bit rate which is higher than the previous bit rate cannot be actually achieved by simply decreasing the quantization step. This is because a current encoding process carried out at a small quantization step on a picture completing a previous encoding process performed at a large quantization step by no means improves the quality of the picture. That is to say, a current encoding process carried out at a small quantization step on a picture completing a previous encoding process performed at a large quantization step merely increases the number of resulting bits, but does not help to improve the quality of the picture. Thus, by using the largest quantization step among the quantization steps used in the current encoding process (or the encoding process of the fourth generation) and the previous encoding processes (or the encoding processes of the first, second and third generations), the most efficient encoding process can be carried out.

[0134] Next, the history decoding apparatus 104 and the history encoding apparatus 107 are explained by referring to Fig. 15. As shown in the figure, the history decoding apparatus 104 comprises a user-data decoder 201, a converter 202 and a history decoder 203. The user-data decoder 201 decodes user data supplied by the decoding apparatus 102. The converter 202 converts data output by the user-data decoder 201 and the history decoder 203 reproduces history information from data output by the converter 202.

[0135] On the other hand, the history encoding apparatus 107 comprises a history formatter 211, a converter 212 and a user-data formatter 213. The history formatter 211 formats encoding parameters of the three generations supplied thereto by the history information separating apparatus 105. The converter 212 converts data output by the history formatter 211 and the user-data formatter 213 formats data output by the converter 212 into a format of the user data.

[0136] The user-data decoder 201 decodes user data supplied by the decoding apparatus 102, supplying a result of the decoding to the converter 202. Details of the user data will be described later. At any rate, the user data denoted by user_data() comprises user_data_start_code and user_data. According to MPEG specifications, generation of 23 consecutive bits of "0" in the user-data is prohibited in order to prevent start_code from being detected incorrectly. Since the history information may include 23 or more consecutive bits of "0", it is necessary to process and convert the history information into converted_history_stream() to be described later by referring to Fig. 38. The component that carries out this conversion by insertion of a "1" bit is the converter 212 employed in the history encoding apparatus 107. On the other hand, the converter 202 employed in the history encoding apparatus 104 carries out conversion of deletion of a bit opposite to the conversion performed by the converter 212 employed in the history encoding apparatus 107.

[0137] The history decoder 203 generates history information from data output by the converter 202, outputting the information to the history information multiplexing apparatus 103.

[0138] On the other hand, the history formatter 211 employed in the history encoding apparatus 107 converts formats encoding parameters of the three generations supplied thereto by the history information separating apparatus 105 into a format of history information. The format of history information can have a fixed length shown in Figs. 40 to 46 to be described later or a variable length shown in Fig. 47 also to be described later.

[0139] The history information formatted by the history formatter 211 is converted by the converter 212 into converted_history_stream() to prevent start_code of user_data() from being detected incorrectly as described above. That is to say, while history information may include 23 or more consecutive bits of "0", generation of 23 consecutive bits of "0" in user_data is prohibited by MPEG specifications. The converter 212 thus converts the history information by insertion of a "1" bit in accordance with the imposed prohibition.

[0140] The user-data formatter 213 adds Data_ID and user_data_stream_code to converted_history_stream() supplied by the converter 212 on the basis of a syntax shown in Fig. 38 to be described later to generate user_data that can be inserted into video_stream, outputting user_data to the encoding apparatus 106.

[0141] Fig. 20 is a block diagram showing a typical configuration of the history formatter 211. As shown in the figure, code-language converter 301 and a code-language converter 305 receive item data and item-number data from the encoding-parameter separating circuit 105. The item data is encoding parameters, that is, encoding parameters transmitted this time as history information. The item-number data is information used for identifying a stream including the encoding parameters. Examples of the item-number data are a syntax name and the name of sequence_header to be described later. The code-language converter 301 converts an encoding parameter supplied thereto into a code language conforming to a specified syntax, outputting the code to a barrel shifter 302. The barrel shifter 302 barrel-shifts the code language supplied thereto by a shift amount corresponding to information supplied thereto by an address generating circuit 306, outputting the shifted code to a switch 303 in byte units. The switch 303, the contact position of which can be changed over by a bit select signal output by the address generating circuit 306, has as many pairs of contact poles as bits supplied by the barrel shifter 302. The switch 303 passes on the code supplied thereto by the barrel shifter 302 to a RAM unit 304 to be stored at a write address specified by the address generating circuit 306. The code language stored in the RAM unit 304 is read out back from a read address specified by the address generating circuit 306 and supplied to the converter 212 provided at the following stage. If necessary, the code

read out from the RAM unit 304 is again supplied to the RAM unit 304 by way of the switch 303 to be stored therein again.

[0142] The code-length converter 305 determines the code length of the encoding parameter from the syntax and the encoding parameter supplied thereto, outputting information on the code length to the address generating circuit 306. The address generating circuit 306 generates the shift amount, the bit select signal, the write address and the read address described above in accordance with the information on the code length received from the code-length converter 305. The shift amount, the bit select signal and the addresses are supplied to the barrel shifter 302, the switch 303 and the RAM unit 304 respectively.

[0143] As described above, the history formatter 211 functions as the so-called variable-length encoder for carrying out a variable-length encoding process on an encoding parameter supplied thereto and for outputting a result of the variable-length encoding process.

[0144] Fig. 22 is a block diagram showing a typical configuration of the converter 212. In this typical configuration, 8-bit data is read out from a read address in a buffer-memory unit 320 provided between the history formatter 211 and the converter 212 and supplied to an 8-bit D-type flip-flop (D-FF) circuit 321 to be held therein. The read address is generated by a controller 326. The 8-bit data read out from the D-type flip-flop circuit 321 is supplied to a stuff circuit 323 and an 8-bit D-type flip-flop circuit 322 to be held therein. The 8-bit data read out from the D-type flip-flop circuit 322 is also supplied to the stuff circuit 323. To put it in detail, the 8-bit data read out from the D-type flip-flop circuit 321 is concatenated with the 8-bit data read out from the D-type flip-flop circuit 322 to form 16-bit parallel data which is then supplied to the stuff circuit 323.

[0145] The stuff circuit 323 inserts the code "1" into a stuff position specified by a signal to produce data with a total of 17 bits which is supplied to a barrel shifter 324. The signal indicating a stuff position is supplied by the controller 326 and the operation to insert the code "1" is called stuffing.

[0146] The barrel shifter 324 barrel-shifts the data supplied thereto by the stuff circuit 323 by a shift amount indicated by a signal received from the controller 326, extracting 8-bit data out off the shifted one. The extracted data is then output to an 8-bit D-type flip-flop circuit 325 to be held therein. The data held in the D-type flip-flop circuit 325 is finally output to the user-data formatter 213 provided at the following stage by way of a buffer-memory unit 327. That is to say, the data is temporarily stored in the buffer-memory unit 327 provided between the converter 212 and the user-data formatter 213 at a write address generated by the controller 326.

[0147] Fig. 23 is a block diagram showing a typical configuration of the stuff circuit 323. In this configuration, the 16-bit data received from the D-type flip-flop circuits 321 and 322 is supplied to contact points a of switches 331-16 to 331-1. Pieces of data supplied to the contact points a of the switches 331-i where i = 0 to 15 are also supplied to contact points c of the switches 331-i. The switches 331-i where i = 1 to 16 are switches adjacent to the switches 331-i where i = 0 to 15 respectively on the MSB side shown in the figure) of the switches 331-i where i = 0 to 15. For example, the thirteenth piece of data from the LSB supplied to the contact point a of the switch 331-13 on the MSB side of the adjacent switch 331-12 is also supplied to the contact point c of the switch 331-12. At the same time, the fourteenth piece of data from the LSB supplied to the contact point a of the switch 331-14 on the MSB side of the adjacent switch 331-14 is also supplied to the contact point c of the switch 331-13.

[0148] However, the contact point a of the switch 331-0 provided on the lower side of the switch 331-1 is open because no switch is provided on the lower side of the switch 331-0 which corresponds to the LSB. In addition, the contact point c of the switch 331-16 provided on the upper side of the switch 331-15 is also open because no switch is provided on the upper side of the switch 331-16 which corresponds to the MSB.

[0149] The data "1" is supplied to contact points b of the switches 331-0 to 331-16. The decoder 332 changes over one of the switches from 331-0 to 331-16 at a stuff position indicated by a staff position signal received from the controller 326 to the contact point b in order to insert the data "1" into the stuff position. The switches 331-0 to 331-16 on the LSB side of the switch at the stuff position are changed over to their contact points c whereas the switches 331 on the MSB side of the switch at the stuff position are changed over to their contact points a.

[0150] Fig. 23 shows an example in which the data "1" is inserted into the thirteenth bit from the LSB side. Thus, in this case, the switches 331-0 to 331-12 are changed over to their contact points c and the switches 331-14 to 331-16 are changed over to their contact points a. The switch 331-13 is changed over to the contact point b thereof.

[0151] With the configuration described above, the converter 212 shown in Fig. 22 converts the 22-bit code into a 23-bit code including the inserted data "1", outputting the 23-bit result of the conversion.

[0152] Fig. 24 is a diagram showing timing of pieces of data output by a variety of portions of the converter 212 shown in Fig. 22. When the controller 326 employed in the converter 212 generates a read address shown in Fig. 24A synchronously with a clock signal for a byte of data, byte data stored at the read address is read out from the buffer-memory unit 320 and temporarily held by the D-type flip-flop circuit 321. Then, data of Fig. 24B read out from the D-type flip-flop circuit 321 is supplied to the stuff circuit 323 and the D-type flip-flop circuit 322 to be held therein. Data of Fig. 24C read out from the D-type flip-flop circuit 322 is concatenated with the data of Fig. 24B read out from the D-

type flip-flop circuit 321 and data obtained as a result of the concatenation shown in Fig. 24D is supplied to the stuff circuit 323.

[0153] As a result, with timing of a read address A1, the first byte D0 of the data of Fig. 24B read out from the D-type flip-flop circuit 321 is supplied to the stuff circuit 323 as a first byte of the data shown in Fig. 24D. Then, with timing of a read address A1, the second byte D1 of the data of Fig. 24B read out from the D-type flip-flop circuit 321 and concatenated with the first byte D0 of the data of Fig. 24C read out from the D-type flip-flop circuit 322 is supplied to the stuff circuit 322 as a second two bytes of the data shown in Fig. 24D. Subsequently, with timing of a read address A3, the third byte D2 of the data of Fig. 24B read out from the D-type flip-flop circuit 321 and concatenated with the second byte D1 of the data of Fig. 24C read out from the D-type flip-flop circuit 322 is supplied to the stuff circuit 322 as a third two bytes of the data shown in Fig. 24D and so on.

[0154] The stuff circuit 323 receives a signal of Fig. 24E indicating a stuff position, into which the data "1" is to be inserted, from the controller 326. The decoder 332 employed in the stuff circuit 323 changes over one of the switches 331-0 to 331-16 at the stuff position to the contact point b thereof. The switches 331 on the LSB side of the switch at the stuff position are changed over to their contact points c whereas the switches 331 on the MSB side of the switch at the stuff position are changed over to their contact points a. As a result, the stuff circuit 323 inserts the data "1" into the stuff position signal, outputting data with the inserted data "1" shown in Fig. 24F.

[0155] The barrel shifter 324 barrel-shifts the data supplied thereto by the stuff circuit 323 by a shift amount indicated by a shift signal of Fig. 24G received from the controller 326, outputting a shifted signal shown in Fig. 24H. The shifted signal is held temporarily in the D-type flip-flop circuit 325 before being output to a later stage as shown in Fig. 24I.

[0156] The data output from the D-type flip-flop circuit 325 includes the data "1" inserted into a position following to 22-bit data. Thus, the number of consecutive 0 bits never exceeds 22 even if all bits between the data "1" and the next data "1" are "0".

[0157] Fig. 25 is a block diagram showing a typical configuration of the converter 202. The fact that components ranging from a D-type flip-flop circuit 341 to a controller 346 are employed in the converter 202 as the components ranging from the D-type flip-flop circuit 321 to the controller 326 are employed in the converter 212 shown in Fig. 22 indicates that the configuration of the former is basically the same as the configuration of the latter. The converter 202 is different from the converter 212 in that, in the case of the former, a delete circuit 343 is employed in place of the stuff circuit 323 of the latter. Otherwise, the configuration of the converter 202 is the same as that of the converter 212 shown in Fig. 22.

[0158] The delete circuit 343 employed in the converter 202 deletes a bit at a delete position indicated by a signal output by the controller 346. The delete position corresponds to the stuff position, into which the stuff circuit 323 shown in Fig. 23 inserts the data "1".

[0159] The remaining operations of the converter 202 are the same as those carried out by the converter 212 shown in Fig. 22.

[0160] Fig. 26 is a block diagram showing a typical configuration of the delete circuit 343. In this configuration, the 15 bits on LSB side of which 16-bit data received from the D-type flip-flop circuits 341 and 342 are supplied to contact points a of switches 351-0 to 351-14. Pieces of data supplied to the contact points a of the switches 351-i where i = 1 to 14 are also supplied to contact points b of the switches 351-i where i = 0 to 13. The switches 351-i where i = 1 to 14 are switches adjacent to the switches 351-i where i = 0 to 13 respectively on the MSB side (or the upper side shown in the figure) of the switches 351-i where i = 0 to 13. For example, the thirteenth piece of data from the LSB supplied to the contact point a of the switch 351-13 on the MSB side of the adjacent switch 351-12 is also supplied to the contact point b of the switch 351-12. At the same time, the 14th piece of data from the LSB supplied to the contact point a of the switch 351-14 on the MSB side of the adjacent switch 351-13 is also supplied to the contact point c of the switch 351-13. The decoder 352 deletes a bit at a delete position indicated by a signal output by the controller 346, outputting the remaining 15-bit data excluding the deleted bit.

[0161] Fig. 26 shows a state in which the thirteenth input bit from the LSB (input bit 12) is deleted. Thus, in this case, the switches 351-0 to 351-11 are changed over to their contact points a to output 12 input bits from the LSB (bit 0) to the twelfth bit (bit 11) as they are. On the other hand, the switches 351-12 to 351-14 are changed over to their contact points b to pass on the fourteenth to sixteenth input bits (input bits 13 to 15) as the thirteenth to fifteenth output bits (output bits 12 to 14) respectively. In this state, the thirteenth input bit from the LSB (input bit 12) is connected to none of the output lines.

[0162] 16-bit data is supplied to the stuff circuit 323 shown in Fig. 23 and the delete circuit 343 shown in Fig. 26. This is because the data supplied to the stuff circuit 323 is a result of concatenation of pieces of data output by the 8-bit D-type flip-flop circuits 321 and 322 employed in the converter 212 shown in Fig. 22. By the same token, the data supplied to the delete circuit 343 is a result of concatenation of pieces of data output by the 8-bit D-type flip-flop circuits 341 and 342 employed in the converter 202 shown in Fig. 25. The barrel shifter 324 employed in the converter 212 shown in Fig. 22 barrel-shifts the 17-bit data supplied thereto by the stuff circuit 323 by a shift amount indicated by a signal received from the controller 326, finally extracting data of typically 8 bits out off the shifted one. Likewise, the barrel shifter 344 employed in the converter 202 shown in Fig

25 barrel-shifts the 15-bit data supplied thereto by the stuff circuit 324 by a shift amount indicated by a signal received from the controller 346, finally extracting data of 8 bits out off the shifted one.

[0163] Fig. 21 is a block diagram showing a typical configuration of the history decoder 203 for decoding data completing the history formatting process in the converter 202. Data of an encoding parameter supplied to the history decoder 203 by the converter 202 is fed to a RAM unit 311 to be stored therein at a write address specified by an address generating circuit 315. The address generating circuit 315 also outputs a read address with predetermined timing to the RAM unit 311. At that time, data stored at the read address in the RAM unit 311 is output to a barrel shifter 312. The barrel shifter 312 barrel-shifts the data supplied thereto by a shift amount corresponding to information supplied thereto by the address generating circuit 315, outputting the shifted data to inverse-code-length converters 313 and 314.

[0164] The inverse-code-length converters 313 and 314 also receive the name of a syntax of a stream including the encoding parameters from the converter 202. The inverse code-length converter 313 determines the code length of the encoding parameter from the data or the code language supplied thereto in accordance with the syntax, outputting information on the code length to the address generating circuit 315.

[0165] On the other hand, the inverse-code-length converter 314 decodes or reversely encodes the data supplied by the barrel shifter 312 on the basis of the syntax, outputting a result of the decoding process to the encoding-parameter multiplexing apparatus 103.

[0166] In addition, the inverse-code-length converter 314 also extracts information required for identifying what code language is included, that is, information required for determining a delimiter of a code, outputting the information to the address generating circuit 315. The address generating circuit 315 generates write and read addresses and a shift amount based on this information and the code length received from the inverse-code-length converter 313, outputting the write/read addresses to the RAM unit 311 and the shift amount to the barrel register 312.

[0167] Fig. 27 is a block diagram showing another typical configuration of the converter 212. A counter 361 employed in this configuration counts the number of consecutive 0 bits of data supplied thereto, outputting the result of counting to the controller 326. When the number of consecutive 0-bit of data supplied to the counter 361 reaches 22, the controller 326 outputs a signal representing a stuff position to the stuff circuit 323. At the same time, the controller 326 resets the counter 361, allowing the counter 361 to start counting the number of consecutive 0-bit again from 0. The rest of the configuration and the operation are the same as those of the converter 212 shown in Fig. 22.

[0168] Fig. 28 is a block diagram showing another typical configuration of the converter 202. A counter 371 employed in this configuration counts the number of consecutive 0-bit of data supplied thereto, outputting the result of counting to the controller 346. When the number of consecutive 0-bit of data supplied to the counter 371 reaches 22, the controller 346 outputs a signal representing a delete position to the delete circuit 343. At the same time, the controller 346 resets the counter 371, allowing the counter 371 to start counting the number of consecutive 0-bit again from 0. The rest of the configuration and the operation are the same as those of the converter 202 shown in Fig. 25.

[0169] As described above, in the typical configurations shown in Figs. 27 and 28, the data "1" is inserted as a marker bit and deleted respectively when a predetermined pattern comprising a predetermined number of consecutive 0-bit is detected by the counter. The typical configurations shown in Figs. 27 and 28 allow processing to be carried out with a higher degree of efficiency than the configurations shown in Figs. 22 and 25 respectively.

[0170] Fig. 29 is a block diagram showing a typical configuration of the user-data formatter 213. In this configuration, when a controller 383 outputs a read address to a buffer memory not shown provided between the converter 212 and the user-data formatter 213, data is output from the read address and supplied to a contact point a of a switch 382 employed in the user-data formatter 213. It should be noted that the buffer memory itself is not shown in the figure. In a ROM unit 381, data required for generating user_data() such as a user-data start code and a data ID is stored. A controller 313 changes over the switch 382 to the contact point a or a contact point b with predetermined timing in order to allow the switch 382 to select the data stored in the ROM unit 381 or data supplied by the converter 212 and pass on the selected data. In this way, data with a format of user_data() is output to the encoding apparatus 106.

[0171] It is worth noting that the user-data decoder 201 can be implemented by outputting input data by way of a switch for deleting inserted data read out from a ROM unit like the ROM unit 381 employed in the user-data formatter 213 shown in Fig. 29. The configuration of the user-data decoder 201 is shown in none of the figures.

[0172] Fig. 30 is a block diagram showing the state of an implementation in which a plurality of transcoders 101-1 to 101-N are connected in series for the use of video editing studio service. The encoding-parameter multiplexing apparatuses 103-i employed in the transcoders 101-i where i = 1 to N each write most recent encoding parameters used by itself over a region for storing least recent encoding parameters in an area used for recording encoding parameters. As a result, base-band picture data includes encoding parameters or generation history information of the four most recent generations associated with the macroblocks of the pic-

ture data.

[0173] The variable-length-encoding circuit 58 employed in the encoder 121-i of Fig. 19 employed in the encoding apparatus 106-i encodes video data received from the quantization circuit 57 on the basis of current encoding parameters received from the encoding-parameter separating circuit 105-i. As a result, the current encoding parameters are multiplexed typically in picture_header() included in a bitstream generated by the variable-length-encoding circuit 58.

[0174] In addition, the variable-length-encoding circuit 58 also multiplexes user data, which includes generation history information and is received from the history encoding apparatus 107-i, into the output bitstream. This multiplexing process is not the embedded processing like the one shown in Fig. 18, but multiplexing of the user data into the bitstream. Then, the bitstream output by the encoding apparatus 106-i is supplied to the transcoder 101-(i+1) at the following stage by way of the SDTI 108-i.

[0175] The configurations of the transcoders 101-i and 101-(i+1) are the same as the one shown in Fig. 15. The processing carried out by them can thus be explained by referring to Fig. 15. If it is desired to change the current picture type from the I-picture to the P- or B-picture in an encoding operation using the history of actual encoding parameters, the history of previous encoding parameters is searched for those of a P- or B-picture used in the previous. If a history of a P or B-picture is found in the history, its parameters including a motion vector are used to change the picture type. If a history of a P- or B-picture is not found in the history, on the other hand, the modification of the picture type without motion detection is given up. It is needless to say that the picture type can be changed even if an encoding parameter of a P- or B-picture is not found in the history provided that motion detection is carried out.

[0176] In the format shown in Fig. 18, encoding parameters of four generations are embedded in picture data. As an alternative, parameters for each of the I-, P- and B-pictures can also be embedded in a format like one shown in Fig. 31. In the example shown in Fig. 31, encoding parameters or picture history information of one generation are recorded for each picture type in an operation to encode the same macroblocks accompanying changes in picture type occurring in the previous. In this case, the decoder 111 shown in Fig. 16 outputs encoding parameters of one generation for the I-, P- and B-pictures to be supplied to the encoder 121 shown in Fig. 19 instead of encoding parameters of most recent, the first, second and third preceding generations.

[0177] In addition, since the area of Cb[1][x] and Cr[1][x] is not used, the present invention can also be applied to picture data of a 4 : 2 : 0 format which does not use the area of Cb[1][x] and Cr[1][x]. In the case of this example, the decoding apparatus 102 fetches encoding parameters in the course of decoding and identifies the picture type. The decoding apparatus 102 writes or multiplexes the encoding parameters into a location corresponding to the picture type of the picture signal and outputs the multiplexed picture signal to the encoding-parameter separating apparatus 105. The encoding-parameter separating apparatus 105 separates the encoding parameters from the picture data and, by using the separated encoding parameters, the encoding-parameter separating apparatus 105 is capable of carrying out a post-decoding-encoding process while changing the picture type by taking a picture type to be changed and previous encoding parameters supplied thereto into consideration.

[0178] The transcoder 101 has another operation which is different from the parameter-reuse-encoding process to deter mine a changeable picture type in only case of the controller 70 does not allow the motion vector detecting circuits to operate.

[0179] The other operation is explained by referring to a flowchart shown in Fig. 32. As shown in Fig. 32, the flowchart begins with a step S1 at which encoding parameters or picture history information of one generation for each picture type are supplied to the controller 70 of the encoder 121. The flow of the processing then goes on to a step S2 at which the encoding-parameter separating apparatus 105 forms a judgment as to whether or not picture history information includes encoding parameters used in a change to a B-picture. If the picture history information includes encoding parameters used in a change to a B-picture, the flow of the processing proceeds to a step S3.

[0180] At the step S3, the econtroller 70 forms a judgment as to whether or not picture history information includes encoding parameters used in a change to a P-picture. If the picture history information includes encoding parameters used in a change to a P-picture, the flow of the processing proceeds to a step S4.

[0181] At the step S4, the controller 70 determines that changeable picture types are the I-, P- and B-pictures. If the outcome of the judgment formed at the step S3 indicates that the picture history information does not include encoding parameters used in a change to a P-picture, on the other hand, the flow of the processing proceeds to a step S5.

[0182] At the step S5, the controller 70 determines that changeable picture types are the I- and B-pictures. In addition, the controller 70 determines that a pseudo-change to a P-picture is also possible by carrying out special processing using only a forward-prediction vector and no backward-prediction vector included in history information of the B-picture. If the outcome of the judgment formed at the step S2 indicates that the picture history information does not include encoding parameters used in a change to a B-picture, on the other hand, the flow of the processing proceeds to a step S6

[0183] At the step S6, the controller 70 forms a judgment as to whether or not picture history information includes encoding parameters used in a change to a P-

picture. If the picture history information includes encoding parameters used in a change to a P-picture, the flow of the processing proceeds to a step S7.

[0184] At the step S7, the controller 70 determines that changeable picture types are the I- and P-pictures. In addition, the encoding-parameter separating apparatus 105 determines that a change to a B-picture is also possible by carrying out special processing using only a forward-prediction vector and no backward-prediction vector included in history information of the P-picture.

[0185] If the outcome of the judgment formed at the step S6 indicates that the picture history information does not include encoding parameters used in a change to a P-picture, on the other hand, the flow of the processing proceeds to a step S8. At the step S8, the controller 70 determines that the only changeable picture type is the I-picture because there is no motion vector. An I-picture can not be changed to any other picture type than the I-picture.

[0186] After completing the steps S4, S5, S7 or S8, the flow of the processing goes on to a step S9 at which the controller 70 notifies the user of the changeable picture types on a display unit which is shown in none of the figures.

[0187] Fig. 33 is a diagram showing examples of changes in picture type. When the picture type is changed, the number of frames composing a GOP structure is changed. To put it in detail, in these examples, a long GOP structure is changed to a short GOP structure of the second generation. And then, the GOP structure of athe second generation is changed back to a long GOP at a third generation. The long GOP structure has an N = 15 and an M = 3, where N is the number of frames constituting the GOP and M is the period of the appearance of the P-picture expressed in terms of frames. On the other hand, the short GOP has an N = 1 and an M = 1 where M is the period of the appearance of the I-picture expressed in terms of frames. It should be noted that a dashed line shown in the figure represents a boundary between two adjacent GOPs.

[0188] When the GOP structure of the first generation is changed to the GOP structure of the second generation, the picture types of all the frames can be changed to the I-picture as is obvious from the explanation of the processing to determine changeable picture types given above. When these picture types are changed, all motion vectors which were processed when the source video sognal was encoded in the first generation are saved or left. Then, when the short GOP structureis changed back to the long GOP structure at the third generation. That is, even if picture types are changed, the motion vector for each type which was saved when the source video signal was encoded at the first generation is re-utilized, allowing a change back to the long GOP structureto be made with deterioration of the picture quality avoided.

[0189] Fig. 34 is a diagram showing another examples of changes in picture type. In the case of these examples, changes are made from a long GOP structure with N = 14 and M = 2 to a short GOP structure with N = 2 and M = 2 at the second generation and then to a short GOP structure with N = 1 and M = 1 and finally to a random GOP with an undetermined frame count N at the fourth generation.

[0190] Also in these examples, a motion vector for each picture type which was processed when the source video sognal was encoded as the first generation is saved untill the fourth generation. As a result, by reutilizing the saved encoding parameters, deterioration of the picture quality can be reduced to a minimum even if the picture types are changed in a complicated manner as shown in Fig. 34. In addition, if the quantization scale of the saved encoding parameters is utilized effectively, an encoding process which entails only little deterioration of the picture quality can be implemented.

[0191] The re-utilization of the quantization scale is explained by referring to Fig. 35. Fig. 35 is a diagram showing a case in which a certain reference frame is always encoded with an I-picture from a first generation to a fourth generation. Only the bit rate is changed from 4 Mbps for the first generation to 18 Mbps for the second generation and then to 50 Mbps for the third generation and finally back to 4 Mbps for the fourth generation.

[0192] When a bit rate of 4 Mbps of bitstream generated at first generation is changed to a bit rate of 18 Mbps at second generation, the picture quality is not improved even a post-decoding-encoding process is carried out at a fine quantization scale accompanying the increase in bit rate. This is because data quantized in the previous at a coarse quantization step is not restored. Thus, quantization at a fine quantization step accompanying a raise in bit rate in the course of processing as shown in Fig. 35 merely increases the amount of information and does not lead to an improvement of the picture quality. For this reason, if control is executed to sustain a coarsest or largest quantization scale used in the previous, the encoding process can be implemented leapt wastefully and most efficiently.

[0193] As described above, when the bit rate is changed, by making use of the previous history of the quantization scale, the encoding process can be implemented most effectively.

[0194] This quantization control processing is explained by referring to a flowchart shown in Fig. 36. As shown in the figure, the flowchart begins with a step S11 at which controller 70 forms a judgment as to whether or not input picture history information includes an encoding parameter of a picture type to be changed from now on. If the outcome of the judgment indicates that the input picture history information includes an encoding parameter of a picture type to be changed, the flow of the processing goes on to a step S12.

[0195] At the step S12, the controller 70 extracts the history_q_scale_code from the encoding parameters in question for comparison included in the picture history

information.

[0196] The flow of the processing then proceeds to a step S13 at which the controller 70 calculates a candidate value of the feedback_q_scale_code obased on a data fullness of the transmission buffer 59.

[0197] The flow of the processing then proceeds to a step S14 at which the controller 70 forms a judgment as to whether or not history_q_scale_code is larger or coarser than feedback_q_scale_code. If the outcome of the judgment indicates that history_q_scale_code is larger or coarser than feedback_q_scale_code, the flow of the processing continues to a step S15.

[0198] At the step S15, the controller 70 supplies history_q_scale_code as a quantization scale to the quantization circuit 57 which then carries out a quantization process by using history_q_scale_code.

[0199] The flow of the processing then proceeds to a step S16 at which the controller 70 forms a judgment as to whether or not all macroblocks included in the frame have been quantized. If the outcome of the judgment indicates that all the macroblocks included in the frame not have been quantized yet, the flow of the processing goes back to the step S13 to carry out the pieces of processing of the steps S13 to S16 repeatedly till all the macroblocks included in the frame are quantized.

[0200] If the outcome of the judgment formed at the step S14 indicates that history_q_scale_code is not larger than feedback_q_scale_code, that is, history_q_scale_code is finer than feedback_q_scale_code, on the other hand, the flow of the processing continues to a step S17.

[0201] At the step S17, the controller 70 supplies feeds back feedback_q_scale_code as a quantization scale to the quantization circuit 57 which then carries out a quantization process by using feedback_q_scale_code.

[0202] If the outcome of the judgment formed at the step S11 indicates that the input picture history information does not include an encoding parameter of a picture type to be changed, on the other hand, the flow of the processing goes on to a step S18.

[0203] At the step S18, the quantization circuit 57 receives the candidate value of the feedback_q_scale_code from the controller 70.

[0204] The flow of the processing then proceeds to a step S19 at which the quantization circuit 57 which carries out a quantization process by using Q_feedback.

[0205] The flow of the processing then proceeds to a step S20 at which controller 70 forms a judgment as to whether or not all macroblocks included in the frame have been quantized. If the outcome of the judgment indicates that all the macroblocks included in the frame not have been quantized yet, the flow of the processing goes back to the step S18 to carry out the pieces of processing of the steps S18 to S20 repeatedly till all the macroblocks included in the frame are quantized.

[0206] The transcoder 101 explained earlier by referring to Fig. 15 supplies previous encoding parameters of the first, second and third generations to the video encoding apparatus 106 by multiplexing these parameters in base-band video data. In the present invention, however, a technology of multiplexing previous encoding parameters in base-band video data is not absolutely required. For example, previous encoding parameters can be transferred by using a transmission line such as a data transfer bus provided separately from that for the base-band video data as shown in Fig. 37.

[0207] The video decoding apparatus 102, the history decoding apparatus 104, the video encoding apparatus 106 and the history encoding apparatus 107 shown in Fig. 37 have entirely the same configurations and functions as the video decoding apparatus 102, the history decoding apparatus 104, the video encoding apparatus 106 and the history encoding apparatus 107 respectively which have been described earlier by referring to Fig. 15.

[0208] The variable-length decoding circuit 112 employed in the video decoding apparatus 102 extracts encoding parameters of the third generation from the sequence layer, the GOP layer, the picture layer, the slice layer and the macroblock layer of the encoded video bitstream ST(3rd) of the third generation, supplying the parameters to the history encoding apparatus 107 and the controller 70 employed in the video encoding apparatus 106. The history encoding apparatus 107 converts the encoding parameters of the third generation supplied thereto into converted_history_stream() which can be described in the user-data area on the picture layer, supplying converted_history_stream() to the variable-length coding circuit 58 employed in the video encoding apparatus 106 as user data.

[0209] In addition, the variable-length decoding circuit 112 also extracts user data (user_data) including previous encoding parameters of the first and second generations from the user-data area on the picture layer of the encoded video bitstream ST(3rd) of the third generation, supplying the user_data to the history decoding apparatus 104 and the variable-length coding circuit 58 employed in the video encoding apparatus 106. The history decoding apparatus 104 extracts the encoding parameters of the first and second generations from a history steam of the user data which is described in the user-data area as converted_history_stream(), supplying the parameters to the controller 70 employed in the video encoding apparatus 106.

[0210] The controller 70 of the video encoding apparatus 106 controls the encoding process carried out by the video encoding apparatus 106 on the basis of the encoding parameters of the first and second generations received from the history decoding apparatus 104 and the encoding parameters of the third generation received from the video decoding apparatus 102.

[0211] In the meantime, the variable-length coding circuit 58 employed in the video encoding apparatus 106 receives the user data (user_data) including encoding

parameters of the first and second generations from the video decoding apparatus 102 and the user data (user_data) including encoding parameters of the third generation from the history encoding apparatus 107, describing these pieces of user_data in the user-data area on the picture layer of an encoded video bitstream of the fourth generation as history information.

[0212] Fig. 38 is a diagram showing a syntax used for decoding an MPEG video stream. The decoder decodes an MPEG bit stream in accordance with this syntax in order to extract a plurality of meaningful data items or meaningful data elements from the bitstream. In the syntax to be explained below, a function and a conditional statement are each represented by a string of normal characters whereas a data element is represented by a string of bold characters. A data item is described by the Mnemonic representing the name of the data item. In some cases, the Mnemonic also indicates the bit length composing the data item and the type of the data item.

[0213] First of all, functions used in the syntax shown in Fig. 38 are explained. A next_start_code() is a function used for searching a bitstream for a start code described in the bitstream. In the syntax shown in Fig. 38, the next_start_code() function is followed by a sequence_header() function and a sequence_extension() function which are laid out sequentially to indicate that the bitstream includes data elements defined by the sequence_header() and sequence_extension() functions. Thus, a start code, a kind of data element described at the beginning of the sequence_header() and sequence_extension() functions, is found by the next_start_code() function from the bitstream in an operation to decode the bitstream. The start code is then used as a reference to further find the sequence_header() and sequence_extension() functions and decode data elements defined by the sequence_header() and sequence_extension() functions.

[0214] It should be noted that the sequence_header() function is a function used for defining header data of a sequence layer in the MPEG bitstream whereas the sequence_extension() function is a function used for defining extension data of a sequence layer in the MPEG bitstream.

[0215] A do {} while statement is described after the sequence_extension() function. The do {} while statement comprises a {} block following a do statement and a while statement following the {} block. Data elements described by functions in the {} block following the do statement are extracted from the bitstream as long as a condition defined by the while statement is true. That is to say, the do {} while syntax defines a decoding process to extract data elements described by functions in the {} block following the do statement from the bitstream as long as the condition defined by the while statement is true.

[0216] A nextbits() used in the while statement is a function used to compare a bit or a string of bits appearing in the bitstream with a data element to be decoded next. In the example of the syntax shown in Fig. 38, the nextbits() function compares a string of bits appearing in the bitstream with sequence_end_code used for indicating the end of a video sequence. The condition defined by the while statement is the to be true if the string of bits appearing in the bitstream does not match sequence_end_code. Thus, the do {} while statement described after the sequence_extension() function indicates that data elements defined by functions in the {} block following the do statement are described in the bitstream as long as sequence_end_code used for indicating the end of a video sequence does not appear in the bitstream.

[0217] After the data elements defined by the sequence_extension() function in the bitstream, data elements defined by an extension_and_user_data(0) function are described. The extension_and_user_data(0) function is a function used for defining extension data and user data of the sequence layer of the MPEG bitstream.

[0218] A do {} while statement following the extension_and_user_data(0) function is a function to extract data elements described by functions in the {} block following the do statement from the bitstream as long as a condition defined by the while statement is true. The nextbits() functions used in the while statement are functions used to form a judgment as to whether or not a bit or a string of bits appearing in the bitstream matches picture_start_code or group_start_code start codes respectively by comparing the string with the start code specified in the function. If the string of bits appearing in the bitstream matches picture_start_code or group_start_code, a condition defined by the while statement is said to be true. Thus, if picture_start_code or group_start_code appears in the bitstream, the codes of data elements defined by functions in the {} block following the do statement are described after this start code. Accordingly, by finding a start code represented by picture_start_code or group_start_code, it is possible to extract data elements defined by functions in the {} block of the do statement from the bitstream.

[0219] An if-statement described at the beginning of the {} block of the do statement states a condition "if group_start_code appears in the bitstream." A true (satisfied) condition stated by the if-statement indicates that data elements defined by a group_of_picture_header(1) function and a extension_and_user_data(1) function are described sequentially after group_start_code.

[0220] The group_of_picture_header(1) function is a function used for defining header data of a GOP layer of the MPEG bitstream and the extension_and_user_data(1) function is a function used for defining extension data named extension_data and/or user data named user_data of the GOP layer of the MPEG bitstream.

[0221] Furthermore, in this bitstream, data elements defined by a picture_header() function and a picture_coding-extension() function are described after the data elements defined by the group_of_picture_header(1) function and the extension_and_user_data(1) function. Of course, if the condition defined by the if-statement is not true, the data elements defined by the group_of_picture_header(1) function and the extension_and_user_data(1) function are not described. In this case, the data elements defined by the picture_header() function and the picture_coding_extension() function are described after the data elements defined by an extension_and_user_data(0) function.

[0222] The picture_header() function is a function used for defining header data to a picture layer of the MPEG stream and the picture_coding_extension() function is a function used for defining first extension data of the picture layer of the MPEG stream.

[0223] The next while statement is a function used for defining a condition. A condition defined by each if-statement described in a {} block following the condition defined by the while statement is judged to be true or false as long as the condition defined by the while statement is true. nextbits() functions used in the while statement are functions for forming a judgment as to whether a string of bits appearing in the bitstream matches extension_start_code and user_start_code respectively. If the string of bits appearing in the bitstream matches extension_start_code or user_data_start, a condition defined by the while statement is said to be true.

[0224] A first if-statement in the {} block following the while statement is a function for forming a judgment as to whether or not a string of bits appearing in the bitstream matches extension_start_code. A string of bits appearing in the bitstream that matches 32-bit extension_start_code indicates that data elements defined by an extension_data(2) function are described after extension_start_code in the bitstream.

[0225] A second if-statement is a function for forming a judgment as to whether or not a string of bits appearing in the bitstream matches user_data_start_code. If a string of bits appearing in the bitstream matches 32-bit user_data_start_code, a condition defined by a third if-statement is judged to be true or false. The user_data_start_code is a start code used for indicating the beginning of a user-data area of the picture layer of the MPEG bitstream.

[0226] The third if-statement in the {} block following the while statement is a function for forming a judgment as to whether or not a string of bits appearing in the bitstream matches History_Data_ID. A string of bits appearing in the bitstream that matches 8-bit History_Data_ID indicates that data elements defined by a converted_history_stream() function are described after a code indicated by 8-bit History_Data_ID in the user-data area of the picture layer of the MPEG bitstream.

[0227] A converted_history_stream() function is a function used for describing history information and history data for transmitting all encoding parameters used in the MPEG encoding process. Details of data elements defined by this converted_history_stream() function will be described later. History_Data_ID is a start code used for indicating the beginning of a description of the history information and the history data in the user-data area of the picture layer of the MPEG bitstream.

[0228] An else statement is syntax indicating a case for a false condition defined by the third if-statement. Thus, if data elements defined by a converted_history_stream() function are not described in the user-data area of the picture layer of the MPEG bitstream, data elements defined by a user_data() function are described.

[0229] A picture_data() function is a function used for describing data elements related to a slice layer and a macroblock layer after user data of the picture layer of the MPEG bitstream. Normally, data elements defined by this picture_data() function are described after data elements defined by a user_data() function or data elements defined by a converted_history_stream() function described in the user-data area of the picture layer of the bitstream. If neither extension_start_code nor user_data_start_code exists in a bitstream showing data elements of the picture layer, however, data elements defined by this picture_data() function are described after data elements defined by a picture_coding_extension() function.

[0230] After the data elements defined by this picture_data() function, data elements defined by a sequence_header() function and a sequence_extension() function are described sequentially. The data elements described by the sequence_header() function and the sequence_extension() function are exactly the same data elements as defined by a sequence_header() function and a sequence_extension() function described at the beginning of a sequence of the video stream. The reason why the same pieces of data are defined in the stream is to prevent data of the sequence layer from being no longer receivable, thus, preventing a stream from being no longer decodable when reception of a bitstream is started by a bitstream receiving apparatus in the middle of a data stream, such as part of bitstream corresponding to a picture layer.

[0231] After the data elements defined by the sequence_header() function and the sequence_extension() function, that is, at the end of the data stream, 32-bit sequence_end_code used for indicating the end of the sequence is described.

[0232] Fig. 39 is a schematic diagram showing an outline of the basic configuration of the syntax described so far.

[0233] Next, a history stream defined by a

converted_history_stream() function is explained.

[0234] The converted_history_stream() function is a function used for inserting a history stream showing history information into the user-data area of the picture layer of the MPEG bitstream. It should be noted that the word 'converted' means that the stream has completed a conversion process to insert one marker bit for at least every 22 bits of the history stream composed of history data to be inserted into the user area in order to avoid start emulation.

[0235] The converted_history_stream() function is described in either one of a format of a fixed-length history stream shown in Figs. 40 to 46 or a variable-length history stream shown in Fig. 47 to be described later. If the fixed-length history stream is selected on the encoder side, there is a merit that a circuit and software employed in the decoder for decoding data elements from the history stream become simple. If the variable-length history stream is selected on the encoder side, on the other hand, the encoder is capable of selecting arbitrarily history information or data elements described in the user area of the picture layer when necessary. Thus, the amount of data of the history stream can be reduced. As a result, the data rate of the bitstream as a whole can also be lowered as well.

[0236] The history information, the history data and the history parameter cited in the explanation of the present invention are encoding parameters or data elements used in the related art encoding processes and are not encoding-parameter data used in the current encoding process or the encoding process carried out at the last stage. Consider a case in which a picture is encoded and transmitted as an I-picture in an encoding process of a first generation, as a P-picture in an encoding process of a second generation and as a B-picture in an encoding process of a third generation. Encoding parameters used in the encoding process of the third generation are described at predetermined locations of the sequence, GOP-picture, slice and macroblock layers of an encoded bitstream generated as a result of the encoding process of the third generation. On the other hand, encoding parameters used in the encoding process of the first and second generations are not recorded in the sequence or GOP layer for recording the encoding parameters used in the encoding process of the third generation. but recorded in the user-data area of the picture layer as history information of the encoding parameters.

[0237] First of all, the syntax of the fixed-length history stream is explained by referring to Figs. 40 to 46.

[0238] In the first place, encoding parameters related to the sequence header of the sequence layer used in the previous encoding processes, that is, the encoding processes of typically the first and second generations, are inserted as a history stream into the user-data area of the picture layer of the bitstream generated in the encoding process carried out at the last stage, that is, the encoding process of typically the third generation. It should be noted that history information related to the sequence header of the sequence layer of the bitstream generated in the previous encoding processes is never inserted into the sequence header of the sequence layer of the bitstream generated in the encoding process carried out at the last stage.

[0239] Data elements related to the sequence header used in the previous encoding processes include sequence_header_code, sequence_header_present_flag, horizontal_size_value, vertical_size_value, aspect_ratio_information, frame_rate_code, bit_rate_value, marker_bit, VBV_buffer_size_value, constrained_parameter_flag, load_intra_quantizer_matrix, intra_quantizer_matrix, load_non_intra_quantizer_matrix and non_intra_quantizer_matrix as shown in Fig 40.

[0240] The data elements listed above are described as follows. The sequence_header_code data element is a start synchronization code of the sequence layer. The sequence_header_present_flag data element is a flag used for indicating whether data in the sequence header is valid or invalid. The horizontal_size_value data element is data comprising the low-order 12 bits of the number of pixels of the picture in the horizontal direction. The vertical_size_value data element is data comprising low-order 12 bits of the number of pixels of the picture in the vertical direction. The aspect_ratio_information data element is an aspect ratio, that is, a ratio of the height to the width of the picture, or the aspect ratio of the display screen. The frame_rate_code data element is data representing the picture display period.

[0241] The bit_rate_value data element is data comprising the low-order 18 bits of a bit rate for limiting the number of generated bits. The data is rounded up in 400-bsp units. The marker_bit data element is bit data inserted for preventing start-code emulation. The VBV_buffer_size_value data element is data comprising the low-order 10 bits of a value for determining the size of a virtual buffer (video buffer verifier) used in control of the amount of generated code. The constrained_parameter_flag data element is a flag used for indicating whether or not parameters are under constraint. The load_intra_quantizer_matrix data element is a flag used for indicating whether or not data of an intra-MB quantization matrix exists. The intra_quantizer_matrix data element is the value of the intra-MB quantization matrix. The load_non_intra_quantizer_matrix data element is a flag used for indicating whether or not data of a non-intra-MB quantization matrix exists. The non_intra_quantizer_matrix data element is the value of the non-intra-MB quantization matrix.

[0242] Subsequently, data elements representing a sequence extension used in the previous encoding processes are described as a history stream in the user area of the picture layer of the bitstream generated in the encoding process carried out at the last stage

[0243] Data elements representing the sequence extension used in the previous encoding include extension_start_code, extension_start_code_identifier, sequence_extension_present_flag, profile_and_level_identification, progressive_sequence, chroma_format, horizontal_size_extension, vertical_size_extension, bit_rate_extension, vbv_buffer_size_extension, low_delay, frame_rate_extension_n and frame_rate_extension_d as shown in Figs. 40 and 41.

[0244] The data elements listed above are described as follows. The extension_start_code data element is a start synchronization code of extension data. The extension_start_code_identifier data element is data used for indicating which extension data is transmitted. The sequence_extension_present_flag data element is a flag used for indicating whether data in the sequence extension is valid or invalid. The profile_and_level_identification data element is data specifying a profile and a level of the video data. The progressive_sequence data element is data showing that the video data has been obtained from sequential scanning. The chroma_format data element is data specifying the color-difference format of the video data.

[0245] The horizontal_size_extension data element is the two high-order bits data to be added to horizontal_size_value of the sequence header. The vertical_size_extension data element is the two high-order bits data to be added to vertical_size_value of the sequence header. The bit_rate_extension data element is the twelve high-order bits data to be added to bit_rate_value of the sequence header. The vbv_buffer_size_extension data element is the eight high-order bits data to be added to vbv_buffer_size_value of the sequence header. The low_delay data element is data used for indicating that a B-picture is not included. The frame_rate_extension_n data element is data used for obtaining a frame rate in conjunction with frame_rate_code of the sequence header. The frame_rate_extension_d data element is data used for obtaining a frame rate in conjunction with frame_rate_code of the sequence header.

[0246] Subsequently, data elements representing a sequence-display extension of the sequence layer used in the previous encoding processes are described as a history stream in the user area of the picture layer of the bitstream.

[0247] Data elements described as a sequence-display extension are extension_start_code, extension_start_code_identifier, sequence_display_extension_present_flag, video_format, color_decription, color_primaries, transfer_characteristics, matrix_coefficients, display_horizontal_size and display_vertical_size as shown in Fig. 41.

[0248] The data elements listed above are described as follows. The extension_start_code data element is a start synchronization code of extension data. The extension_start_code_identifier data element is data used for indicating which extension data is transmitted. The sequence_display_extension_presentation_flag data element is a flag used for indicating whether data elements in the sequence extension are valid or invalid. The video_format data element is data representing the video format of the source signal. The color_description data element is data used for indicating that detailed data of a color space exists. The color_primaries data element is data showing details of a color characteristic of the source signal. The transfer_characteristics data element is data showing details of how opto-electrical conversion has been carried out. The matrix_coefficients data element is data showing details of how a source signal has been converted from the three primary colors of light. The display_horizontal_size data element is data representing the activation area or the horizontal size of an intended display. The display_vertical_size data element is data representing the activation area or the vertical size of the intended display.

[0249] Subsequently, macroblock assignment data (named macroblock_assignment_in_user_data) showing phase information of a macroblock generated in the previous encoding processes is described as a history stream in the user area of the picture layer of a bitstream generated in the encoding process carried out at the last stage.

[0250] The macroblock_assignment_in_user_data showing phase information of a macroblock comprises data elements such as macroblock_assignment_present_flag, v_phase and h_phase as shown in Fig. 41.

[0251] The data elements listed above are described as follows. The macroblock_assignment_present_flag data element is a flag used for indicating whether data elements of macroblock_assignment_in_user_data are valid or invalid. The v_phase data element is data showing phase information in the vertical direction which is obtained when the macroblock is detached from picture data. The h_phase data element is data showing phase information in the horizontal direction which is obtained when the macroblock is detached from picture data.

[0252] Subsequently, data elements representing a GOP header of the GOP layer used in the previous encoding processes are described as a history stream in the user area of the picture layer of a bitstream generated in the encoding process carried out at the last stage.

[0253] The data elements representing the GOP header are group_start_code, group_of_picture_header_present_flag, time_code, closed_gop and broken_link as shown in Fig. 41.

[0254] The data elements listed above are described as follows. The group_start_code data element is the start synchronization code of the GOP layer. The group_of_picture_header_present_flag data element is

a flag used for indicating whether data elements in group_of_picture_header are valid or invalid. The time_code data element is a time code showing the length of time measured from the beginning of the first picture of the GOP. The closed_gop data element is a flag used for indicating whether or not it is possible to carry out an independent playback operation of a picture in one GOP from another GOP. The broken_link data element is a flag used for indicating whether or not the B-picture at the beginning of the GOP can not be reproduced with a high degree of accuracy because of reasons such as editing.

[0255] Subsequently, data elements representing a picture header of the picture layer used in the previous encoding processes are described as a history stream in the user area of the picture layer of a bitstream generated in the encoding process carried out at the last stage.

[0256] The data elements related to a picture header are picture_start_code, temporal_reference, picture_coding_type, vbv_delay, full_pel_forward_vector, forward_f_code, full_pel_backward_vector and backward_f_code as shown in Figs. 41 and 42.

[0257] The data elements listed above are described concretely as follows. The picture_start_code data element is the start synchronization code of the picture layer. The temporal_reference data element is a number used for indicating a display order of the picture. This number is reset at the beginning of the GOP. The picture_coding_type data element is data used for indicating the type of the picture. The vbv_delay data element is data showing an initial state of a virtual buffer at a random access. The full_pel_forward_vector data element is a flag used for indicating whether the precision of the forward motion vector is expressed in terms of pixel units or half-pixel units. The forward_f_code data element is data representing a forward-motion-vector search range. The full_pel_backward_vector data element is a flag used for indicating whether the precision of the backward motion vector is expressed in terms of pixel units or half-pixel units. The backward_f_code data element is data representing a backward-motion-vector search range.

[0258] Subsequently, data elements representing a picture-coding extension of the picture layer used in the previous encoding processes are described as a history stream in the user area of the picture layer of a bitstream generated in the encoding process carried out at the last stage.

[0259] The data elements related to the picture-coding extension are extension_start_code, extension_start_code_identifier, f_code[0][0], f_code[0][1], f_code[1][0], f_code[1][1], intra_dc_precision, picture_structure, top_field_first, frame_predictive_frame_dct, concealment_motion_vectors, q_scale_type, intra_vlc_format, alternate_scan, repeat_first_field, chroma_420_type, progressive_frame, composite_display_flag, v_axis, field_sequence, sub_carrier, burst_amplitude and sub_carrier_phase as shown in Fig. 42.

[0260] The data elements listed above are described as follows. The extension_start_code data element is a start code used for indicating the start of extension data of the picture layer. The extension_start_code_identifier data element is a code used for indicating which extension data is transmitted. The f_code[0][0] data element is data representing a horizontal motion-vector search range in the forward direction. The f_code[0][1] data element is data representing a vertical motion-vector search range in the forward direction. The f_code[1][0] data element is data representing a horizontal motion-vector search range in the backward direction. The f_code[1][1] data element is data representing a vertical motion-vector search range in the backward direction.

[0261] The intra_dc_precision data element is data representing the precision of DC coefficients. The picture_structure data element is data used for indicating whether the data structure is a frame structure or a field structure. In the case of the field structure, the picture_structure data element also indicates whether the field structure is the high-order field or the low-order field. The top_field_first data element is data used for indicating whether the first field of a frame structure is the high-order field or the low-order field. The frame_predictive_frame_dct data element is data used for indicating that the prediction of frame-mode DCT is carried out only in the frame-DCT mode in the case of a frame structure. The concealment_motion_vectors data element is data used for indicating that the intra-macroblock includes a motion vector for concealing a transmission error.

[0262] The q_scale_type data element is data used for indicating whether to use a linear quantization scale or a non-linear quantization scale. The intra_vlc_format data element is data used for indicating whether or not another 2-dimensional VLC is used in the intra-macroblock. The alternate_scan data element is data representing selection to use a zigzag scan or an alternate scan. The repeat_first_field data element is data used in the case of a 2 : 3 pull-down. The chroma_420_type data element is data equal to the value of the next progressive_frame data element in the case of a 4 : 2 : 0 signal format or 0 otherwise. The progressive_frame data element is data used for indicating whether or not this picture has been obtained from sequential scanning. The composite_display_flag data element is a flag used for indicating whether or not the source signal is a composite signal.

[0263] The v_axis data element is data used in the case of a PAL source signal. The field_sequence data element is data used in the case of a PAL source signal. The sub_carrier data element is data used in the case of a PAL source signal. The burst_amplitude data element is data used in the case of a PAL source signal.

The sub_carrier_phase data element is data used in the case of a PAL source signal.

[0264] Subsequently, a quantization-matrix extension used in the previous encoding processes is described as a history stream in the user area of the picture layer of the bitstream generated in the encoding process carried out at the last stage.

[0265] Data elements related to the quantization-matrix extension are extension_start_code, extension_start_code_identifier, quant_matrix_extension_present_flag, load_intra_quantizer_matrix, intra_quantizer_matrix[64], load_non_intra_quantizer_matrix, non_intra_quantizer_matrix[64], load_chroma_intra_quantizer_matrix, chroma_non_intra_quantizer_matrix[64], load_chroma_intra_quantizer_matrix and chroma_non_intra_quantizer_matrix[64] as shown in Fig. 43.

[0266] The data elements listed above are described as follows. The extension_start_code data element is a start code used for indicating the start of the quantization-matrix extension. The extension_start_code_identifier data element is a code used for indicating which extension data is transmitted. The quant_matrix_extension_present_flag data element is a flag used for indicating whether data elements of the quantization-matrix extension are valid or invalid. The load_intra_quantizer_matrix data element is data used for indicating whether or not quantization-matrix data for an intra-macroblock exists. The intra_quantizer_matrix data element is data representing values of a quantization-matrix for an intra-macroblock.

[0267] The load_non_intra_quantizer_matrix data element is data used for indicating whether or not quantization-matrix data for a non-intra-macroblock exists. The non_intra_quantizer_matrix data element is data representing values of a quantization-matrix for a non-intra-macroblock. The load_chroma_intra_quantizer_matrix data element is data used for indicating whether or not quantization-matrix data for a color-difference intra-macroblock exists. The chroma_intra_quantizer_matrix data element is data representing values of a quantization-matrix for a color-difference intra-macroblock. The load_chroma_non_intra_quantizer_matrix data element is data used for indicating whether or not quantization-matrix data for a color-difference non-intra-macroblock exists. The chroma_non_intra_quantizer_matrix data element is data representing values of a quantization-matrix for a color-difference non-intra-macroblock.

[0268] Subsequently, a copyright extension used in the previous encoding processes is described as a history stream in the user area of the picture layer of the bitstream generated in the encoding process carried out at the last stage.

[0269] Data elements related to the copyright extension are extension_start_code extension_start_code_identifier, copyright_extension_present_flag, copyright_flag, copyright_identifier, original_or_copy, copyright_number_1, copy_right_number_2 and copyright_number_3 as shown in Fig. 43.

[0270] The data elements listed above are described as follows. The extension_start_code data element is a start code used for indicating the start of the copyright extension. The extension_start_code_identifier data element is a code used for indicating which extension data is transmitted. The copyright_extension_present_flag data element is a flag used for indicating whether data-elements of the copyright extension are valid or invalid The copyright_flag data element is a flag used for indicating whether or not a copyright has been given to encoded video data in a range up to the next copyright extension or the end of the sequence.

[0271] The copyright_identifier data element is data used for identifying an institution cataloging the copyright specified by the ISO/IEC JTC/SC29. The original_or_copy data element is a flag used for indicating whether data of the bitstream is original or copied data. The copyright_number_1 data element indicates bits 44 to 63 of a copyright number. The copyright_number_2 data element indicates bits 22 to 43 of the copyright number. The copyright_number_3 data element indicates bits 0 to 21 of the copyright number.

[0272] Subsequently, a picture-display extension (picture_display_extension) used in the previous encoding processes is described as a history stream in the user area of the picture layer of the bitstream generated in the encoding process carried out at the last stage.

[0273] Data elements representing the picture-display extension are extension_start_code, extension_start_code_identifier, picture_display_extension_present_flag, frame_center_horizontal_offset_1, frame_center_vertical_offset_1, frame_center_horizontal_offset_2, frame_center_vertical_offset_2, frame_center_horizontal_offset_3 and frame_center_vertical_offset_3 as shown in Fig. 44.

[0274] The data elements listed above are described as follows. The extension_start_code data element is a start code used for indicating the start of the picture-display extension. The extension_start_code_identifier data element is a code used for indicating which extension data is transmitted. The picture_display_extension_present_flag data element is a flag used for indicating whether data elements of the picture-display extension are valid or invalid. The frame_center_horizontal_offset data element is an offset of the display area in the horizontal direction and the frame_center_vertical_offset data element is an offset of the display area in the vertical direction Up to three

offset value of horizontal and vertical offsets can be defined respectively.

[0275] User data is described as a history stream after the history information representing the picture-display extension already explained in the user area of the picture layer of the bitstream generated in the encoding process carried out at the last stage as shown in Fig. 44.

[0276] Following to the user data, information on a macroblock used in the previous encoding processes is described as a history stream as shown in Figs. 44 to 46.

[0277] Information on the macroblock comprises data elements related to the position of the macroblock, data elements related to the mode of the macroblock, data elements related to control of the quantization step, data elements related to motion compensation, data elements related to the pattern of the macroblock and data elements related to the amount of generated code as shown in Figs. 44 to 46. The data elements related to the position of the macroblock include such as macroblock_address_h, macroblock_address_v, slice_header_present_flag and skipped_macroblock_flag. The data elements related to the mode of the macroblock include such as macroblock_quant, macroblock_motion_forward, macroblock_motion_backward, macroblock_pattern, macroblock_intra, spatial_temporal_weight_code_flag, frame_motion_type and dct_type. The data elements related to control of the quantization step includes such as quantiser_scale_code. The data elements related to motion compensation include PMV[0][0][0], PMV[0][0][1], motion_vertical_field_select[0][0], PMV[0][1][0], PMV[0][1][1], motion_vertical_field_select[0][1], PMV[1][0][0], PMV[1][0][1], motion_vertical_field_select[1][0], PMV[1][1][0], PMV[1][1][1] and motion_vertical_field_select[1][1]. The data elements related to the pattern of the macroblock include such as coded_block_pattern, and the data elements related to the amount of generated code are num_mv_bits, num_coef_bits and num_other_bits or the like.

[0278] The data elements related to the macroblock are described in detail as follows.

[0279] The macroblock_address_h data element is data defining the present absolute position of the macroblock in horizontal direction. The macroblock_address_v data element is data defining the present absolute position of the macroblock in vertical direction. The slice_header_present_flag data element is a flag used for indicating this macroblock is located at the beginning of a slice layer, and whether or not being accompanied by a slice header. The skipped_macroblock_flag data element is a flag used for indicating whether or not to skip this macroblock in a decoding process.

[0280] The macroblock_quant data element is data derived from macroblock_type shown in Figs. 65 to 67. This data element indicates whether or not quantiser_scale_code appears in the bitstream. The macroblock_motion_forward data element is data derived from the macroblock_type shown in Figs. 65 to 67 and used in the decoding process. The macroblock_motion_backward data element is data derived from the macroblock_type shown in Figs. 65 to 67 and used in the decoding process. The macroblock_pattern data element is data derived from the macroblock_type shown in Figs. 65 to 67 and it indicates whether or not coded_block_pattern appears in the bitstream.

[0281] The macroblock_intra data element is data derived from the macroblock_type shown in Figs. 65 to 67 and used in the decoding process. The spatial_temporal_weight_code_flag data element is a flag derived from the macrobloc_type shown in Figs. 65 to 67 and used for indicating whether or not spatial_temporal_weight_code showing an up-sampling technique of a low-order layer picture with time scalability exists in the bitstream.

[0282] The frame_motion_type data element is a 2-bit code used for indicating the prediction type of the macroblock of a frame. A frame_motion_type value of "00" indicates that there are two prediction vectors and the prediction type is a field-based prediction type. A frame_motion_type value of "01" indicates that there is one prediction vector and the prediction type is a field-based prediction type. A frame_motion_type value of "10" indicates that there is one prediction vector and the prediction type is a frame-based prediction type. A frame_motion_type value of "11" indicates that there is one prediction vector and the prediction type is a dual-prime prediction type. The field_motion_type data element is a 2-bit code showing the motion prediction of the macroblock of a field. A field_motion_type value of "01" indicates that there is one prediction vector and the prediction type is a field-based prediction type. A field_motion_type value of "10" indicates that there is two prediction vectors and the prediction type is a 18 × 8 macroblock-based prediction type. A field_motion_type value of "11" indicates that there is one prediction vector and the prediction type is a dual-prime prediction type. The dct_type data element is data used for indicating whether the DCT is carried out in the frame-DCT mode or the field-DCT mode. The quantiser_scale_code data element indicates the quantization-step size of the macroblock.

[0283] Next, data elements related to a motion vector are described. In order to reduce the magnitude of a motion vector required in a decoding process, a particular motion vector is subjected to an encoding process by actually encoding a difference between the particular motion vector and a motion vector decoded earlier. A decoder for decoding a motion vector has to sustain four motion-vector prediction values, each of which comprises horizontal and vertical components. These motion-vector prediction values are represented by PMV[r][s][v]. The subscript [r] is a flag used for indicat-

ing whether the motion vector in a macroblock is the first or second vector. To be more specific, an [r] value of "0" indicates the first vector and an [r] value of "1" indicates the second vector. The subscript [s] is a flag used for indicating whether the direction of the motion vector in the macroblock is the forward or backward direction. To be more specific, an [s] value of "0" indicates the forward direction of the motion vector and an [r] value of "1" indicates the backward direction of the motion vector. The subscript [v] is a flag used for indicating whether the component of the motion vector in the macroblock is a component in the horizontal or vertical direction. To be more specific, a [v] value of "0" indicates the horizontal component of the motion vector and a [v] value of "1" indicates the vertical component of the motion vector.

[0284] Thus, PMV[0][0][0] is data representing the horizontal component, the forward motion vector of the first vector. PMV[0][0][1] is data representing the vertical component of the forward motion vector of the first vector. PMV[0][1][0] is data representing the horizontal component of the backward motion vector of the first vector. PMV[0][1][1] is data representing the vertical component of the backward motion vector of the first vector. PMV[1][0][0] is data representing the horizontal component of the forward motion vector of the second vector. PMV[1][0][1] is data representing the vertical component of the forward motion vector of the second vector. PMV[1][1][0] is data representing the horizontal component of the backward motion vector of the second vector. PMV[1][1][1] is data representing the vertical component of the backward motion vector of the second vector.

[0285] A motion_vertical_field_select[r][s] is data used for indicating which referenced field of the prediction format is used. To be more specific, a motion_vertical_field_select[r][s]-value of "0" indicates the top referenced field and a motion_vertical_field_select[r][s]-value of "1" indicates the bottom referenced field to be used.

[0286] In motion_vertical_field_select[r][s], the subscript [r] is a flag used for indicating whether the motion vector in a macroblock is the first or second vector. To be more specific, an [r] value of "0" indicates the first vector and an [r] value of "1" indicates the second vector. The subscript [s] is a flag used for indicating whether the direction of the motion vector in the macroblock is the forward or backward direction. To be more specific, an [s] value of "0" indicates the forward direction of the motion vector and an [r] value of "1" indicates the backward direction of the motion vector. Thus, motion_vertical_field_select[0][0] indicates the referenced field used in the generation of the forward motion vector of the first vector. motion_vertical_field_select[0][1] indicates the referenced field used in the generation of the backward motion vector of the first vector. motion_vertical_field_select[1][0] indicates the referenced field used in the generation of the forward motion vector of the second vector. motion_vertical_field_select[1][1] indicates the referenced field used in the generation of the backward motion vector of the second vector.

[0287] The coded_block_pattern data element is variable-length data used for indicating which DCT block among a plurality of DCT blocks each for storing a DCT coefficient contains a meaningful or non-zero DCT coefficient. The num_mv_bits data element is data representing the amount of code of the motion vector in the macroblock. The num_coef_bits data element is data representing the amount of code of the DCT coefficient in the macroblock. The num_other_bits data element shown in Fig. 46 is data representing the amount of code in the macroblock other than the motion vector and the DCT coefficient.

[0288] Next, a syntax for decoding data elements from a history stream with a variable length is explained by referring to Figs. 47 to 64.

[0289] As shown in Fig. 47, the history stream with a variable length comprises data elements defined by a next_start_code() function, a sequence_header() function, a sequence_exension() function, an extension_and_user_data(0) function, a group_of_picture_header() function, an extension_and_user_data(1) function, a picture_header() function, a picture_coding_extension() function, an extension_and_user_data(2) function and a picture_data() function.

[0290] Since the next_start_code() function is a function used for searching a bitstream for a start code, data elements defined by the sequence_header() function and used in the previous encoding processes are described at the beginning of the history stream as shown in Fig. 48.

[0291] Data elements defined by the sequence_header() function include sequence_header_code, sequence_header_present_flag, horizontal_size_value, vertical_size_value, aspect_ratio_information, frame_rate_code, bit_rate_value, marker_bit, VBV_buffer_size_value, constrained_parameter_flag, load_intra_quantizer_matrix, intra_quantizer_matrix, load_non_intra_quantizer_matrix and non_intra_quantizer_matrix as shown in Fig. 48.

[0292] The data elements listed above are described as follows. The sequence_header_code data element is the start synchronization code of the sequence layer. The sequence_header_present_flag data element is a flag used for indicating whether data in sequence_header is valid or invalid. The horizontal_size_value data element is data comprising the low-order 12 bits of the number of pixels of the picture in the horizontal direction. The vertical_size_value data element is data comprising the low-order 12 bits of the number of pixels of the picture in the vertical direction. The aspect_ratio_information data element is an aspect ratio of pixels of a picture, that is, a ratio of the

height to the width of the picture, or the aspect ratio of the display screen. The frame_rate_code data element is data representing the picture display period. The bit_rate_value data element is data comprising the low-order 18 bits of a bit rate for limiting the number of generated bits. The data is rounded up in 400-bsp units.

[0293] The marker_bit data element is bit data inserts for preventing start-code emulation. The VBV_buffer_size_value data element is data comprising the low-order 10 bits of a value for determining the size a virtual buffer (video buffer verifier) used in control of the amount of generated code. The constrained_parameter_flag data element is a flag used for indicating whether or not parameters are under constraint. The load_intra_quantizer_matrix data element is a flag used for indicating whether or not data of an intra-MB quantization matrix exists. The intra_quantizer_matrix data element is the value of the intra-MB quantization matrix. The load_non_intra_quantizer_matrix data element is a flag used for indicating whether or not data of a non-intra-MB quantization matrix exists. The non_intra_quantizer_matrix data element is the value of the non-intra-MB quantization matrix.

[0294] Following to the data elements defined by the sequence_header() function, data elements defined by the sequence_extension() function are described as a history stream as shown in Fig. 49.

[0295] The data elements defined by the sequence_extension () function include extension_start_code, extension_start_code_identifier, sequence_extension_present_flag, profile_and_level_identification, progressive_sequence, chroma_format, horizontal_size_extension, vertical_size_extension, bit_rate_extension, vbv_buffer_size_extension, low_delay, frame_rate_extension_n and frame_rate_extension_d as shown in Figs. 49.

[0296] The data elements listed above are described as follows. The extension_start_code data element is a start synchronization code of extension data. The extension_start_code_identifier data element is data used for indicating which extension data is transmitted. The sequence_extension_present_flag data element is a flag used for indicating whether data in the sequence extension is valid or invalid. The profile_and_level_identification data element is data specifying a profile and a level of the video data. The progressive_sequence data element is data showing that the video data has been obtained from sequential scanning. The chroma_format data element is data specifying the color-difference format of the video data. The horizontal_size_extension data element is data to be added to horizontal_size_value of the sequence header as the two high-order bits. The vertical_size_extension data element is data to be added to vertical_size_value of the sequence header as the two high-order bits. The bit_rate_extension data element is data to be added to bit_rate_value of the sequence header as the 12 high-order bits. The vbv_buffer_size_extension data element is data to be added to vbv_buffer_size_value of the sequence header as the 8 high-order bits.

[0297] The low_delay data element is data used for indicating that a B-picture is not included. The frame_rate_extension_n data element is data used for obtaining a frame rate in conjunction with frame_rate_code of the sequence header. The frame_rate_extension_d data element is data used for obtaining a frame rate in conjunction with frame_rate_code of the sequence header.

[0298] Following to the data elements defined by the sequence_extension() function, data elements defined by the extension_and_user_data(0) function are described as a history stream as shown in Fig. 50. For (i) with a value other than 2, the extension_and_user_data(i) function describes only data elements defined by a user_data() function as a history stream instead of describing data elements defined by the extension_data() function. Thus, the extension_and_user_data(0) function describes only data elements defined by the user_data() function as a history stream.

[0299] The user_data() function describes user data as a history stream on the basis of a syntax like one shown in Fig. 51.

[0300] Following to the data elements defined by the extension_and_user_data(0) function, data elements defined by the group_of_picture_header() function shown in Fig. 52 and data elements defined by the extension_and_user_data(1) function shown in Fig. 50 are described as a history stream. It should be noted, however, that the data elements defined by the group_of_picture_header() function and data elements defined by the extension_and_user_data(1) function are described only if group_start_code representing the start code of the GOP layer is described in the history stream.

[0301] As shown in Fig. 52, the data elements defined by the group_of_picture_header() function are group_start_code, group_of_picture_header_present_flag, time_code, closed_gop and broken_link.

[0302] The data elements listed above are described as follows. The group_start_code data element is the start synchronization code of the GOP layer. The group_of_picture_header_present_flag data element is a flag used for indicating whether data elements in group_of_picture_header are valid or invalid. The time_code data element is a time code showing the length of time measured from the beginning of the first picture of the GOP. The closed_gop data element is a flag used for indicating whether or not it is possible to carry out an independent playback operation of a picture in the GOP from another GOP. The broken_link data element is a flag used for indicating that the B-pic-

ture at the beginning of the GOP can not be reproduced with a high degree of accuracy because of reasons such as editing.

[0303] Much like the extension_and_user_data(0) function shown in Fig. 50, the extension_and_user_data(1) function describes only data elements defined by the user_data() function as a history stream.

[0304] If group_start_code representing the start code of the GOP layer is not described in the history stream, the data elements defined by the group_of_picture_header() function and data elements defined by the extension_and_user_data(1) function are also not described in the history stream. In this case, data elements defined by the picture_header() function are described after the data elements defined by the extension_and_user_data(0) function.

[0305] The data elements defined by the picture_headr() function are picture_start_code, temporal_reference, picture_coding_type, vbv_delay, full_pel_forward_vector, forward_f_code, full_pel_backward_vector, backward_f_code, extra_bit_picture and extra_information_picture as shown in Fig. 53.

[0306] The data elements listed above are described concretely as follows. The picture_start_code data element is the start synchronization code of the picture layer. The temporal_reference data element is a number used for indicating a display order of the picture. This number is reset at the beginning of the GOP. The picture_coding_type data element is data used for indicating the type of the picture. The vbv_delay data element is data showing an initial state of a virtual buffer at a random access. The full_pel_forward_vector data element is a flag used for indicating whether the precision of the forward motion vector is expressed in terms of integral pixel units or half-pixel units. The forward_f_code data element is data representing a forward-motion-vector search range. The full_pel_backward_vector data element is a flag used for indicating whether the precision of the backward motion vector is expressed in terms of integral pixel units or half-pixel units. The backward_f_code data element is data representing a backward-motion-vector search range. The extra_bit_picture data element is a flag used for indicating whether or not following additional information exists. To be more specific, extra_bit_picture having a value of "0" indicates that no following additional information exists while extra_bit_picture having a value of "1" indicates that following additional information exists. The extra_information_picture data element is information reserved by specifications.

[0307] Following to the data elements defined by the picture_headr() function, data elements defined by the picture_coding_extension() function shown in Fig. 54 are described as a history stream.

[0308] The data elements defined by the picture_coding_extension() function are extension_start_code, extension_start_code_identifier, f_code[0][0], f_code[0][1], f_code[1][0], f_code[1][1], intra_dc_precision, picture_structure, top_field_first, frame_predictive_frame_dct, concealment_motion_vectors, q_scale_type, intra_vlc_format, alternate_scan, repeat_first_field chroma_420_type, progressive_frame, composite_display_flag, v_axis, field_sequence, sub_carrier, burst_amplitude and sub_carrier_phase as shown in Fig. 54.

[0309] The data elements listed above are described as follows. The extension_start_code data element is a start code used for indicating the start of extension data of the picture layer. The extension_start_code_identifier data element is a code used for indicating which extension data is transmitted. The f_code[0][0] data element is data representing a horizontal motion-vector search range in the forward direction. The f_code[0][1] data element is data representing a vertical motion-vector search range in the forward direction. The f_code[1][0] data element is data representing a horizontal motion-vector search range in the backward direction. The f_code[1][1] data element is data representing a vertical motion-vector search range in the backward direction. The intra_dc_precision data element is data representing the precision of DC coefficients.

[0310] The picture_structure data element is data used for indicating whether the data structure is a frame structure or a field structure. In the case of the field structure, the picture_structure data element also indicates whether the field structure is the high-order field or the low-order field. The top_field_first data element is data used for indicating whether the first field of a frame structure is the high-order field or the low-order field. The frame_predictive_frame_dct data element is data used for indicating that the prediction of frame-mode DCT is carried out only in the frame mode in the case of a frame structure. The concealment_motion_vectors data element is data used for indicating that the intra-macroblock includes a motion vector for concealing a transmission error. The q_scale_type data element is data used for indicating whether to use a linear quantization scale or a non-linear quantization scale. The intra_vlc_format data element is data used for indicating whether or not another 2-dimensional VLC is used in the intra-macroblock.

[0311] The alternate_scan data element is data representing selection to use a zigzag scan or an alternate scan. The repeat_first_field data element is data used in the case of a 2 : 3 pull-down. The chroma_420_type data element is data equal to the value of the next progressive_frame data element in the case of a 4 : 2 : 0 signal format or 0 otherwise. The progressive_frame data element is data used for indicating whether or not this picture has been obtained from sequential scanning. The composite_display_flag data element is data used for indicating whether or not the source signal is a

composite signal. The v_axis data element is data used in the case of a PAL source signal. The field_sequence data element is data used in the case of a PAL source signal. The sub_carrier data element is data used in the case of a PAL source signal. The burst_amplitude data element is data used in the case of a PAL source signal. The sub_carrier_phase data element is data used in the case of a PAL source signal.

[0312]. Following to the data elements defined by the picture_coding_extension() function, data elements defined by the extension_and_user_data(2) function shown in Fig. 50 are described as a history stream. It should be noted, however, that data elements defined by the extension_data() function are described by the extension_and_user_data(2) function only if extension_start_code representing the start code of the extension exists in the bit stream. In addition, data elements defined by the user_data() function are described by the extension_and_user_data(2) function after the data elements defined by the extension_data() function only if user_data_start_code representing the start code of the user data exists in the bitstream as shown in as shown in Fig. 50. That is to say, if neither the start code of the extension nor the start code of the user data exists in the bitstream, data elements defined by the extension_data() function and data elements defined by the user_data() function are not described in the bitstream.

[0313] The extension_data() function is a function used for describing a data element representing extension_start_code and data elements defined by a quant_matrix_extension() function, a copyright_extension() function and a picture_display_extension() function as a history stream in the bitstream as shown in Fig. 55.

[0314] Data elements defined by the quant_matrix_extension() function are extension_start_code, extension_start_code_identifier, quant_matrix_extension_present_flag, load_intra_quantizer_matrix, intra_quantizer_matrix[64], load_non_intra_quantizer_matrix, non_intra_quantizer_matrix[64], load_chroma_intra_quantizer_matrix, chroma_intra_quantizer_matrix[64], load_chroma_non_intra_quantizer_matrix and chroma_non_intra_quantizer_matrix[64] as shown in Fig. 56.

[0315] The data elements listed above are described as follows. The extension_start_code data element is a start code used for indicating the start of the quantization-matrix extension. The extension_start_code_identifier data element is a code used for indicating which extension data is transmitted The quant_matrix_extension_present_flag data element is a flag used for indicating whether data elements of the quantization_matrix extension are valid or invalid. The load_intra_quantizer_matrix data element is data used for indicating whether or not quantization-matrix data for an intra-macroblock exists. The intra_quantizer_matrix data element is data representing values of a quantization-matrix for an intra-macroblock.

[0316] The load_non_intra_quantizer_matrix data element is data used for indicating whether or not quantization-matrix-data for a non-intra-macroblock exists. The non_intra_quantizer_matrix data element is data representing values of a quantization-matrix for a non-intra-macroblock. The load_chroma_intra_quantizer_matrix data element is data used for indicating whether or not quantization-matrix data for a color-difference intra-macroblock exists. The chroma_intra_quantizer_matrix data element is data representing values of a quantization-matrix for a color-difference intra-macroblock. The load_chroma_non_intra_quantizer_matrix data element is data used for indicating whether or not quantization-matrix data for a color-difference non-intra-macroblock exists. The chroma_non_intra_quantizer_matrix data element is data representing values of a quantization-matrix for a color-difference non-intra-macroblock.

[0317] The data elements defined by the copyright_extension() function are extension_start_code, extension_start_code_identifier, copyright_extension_present_flag, copyright_flag, copyright_identifier, original_or_copy, copyright_number_1, copy_right_number_2 and copyright-number_3 as shown in Fig. 57.

[0318] The data elements listed above are described as follows. The extension_start_code data element is a start code used for indicating the start of the copyright extension. The extension_start_code_identifier data element is a code used for indicating which extension data is transmitted. The copyright_extension_present_flag data element is a flag used for indicating whether data elements of the copyright extension are valid or invalid.

[0319] The copyright-flag data element is a flag used for indicating whether or not a copyright has been given to encoded video data in a range up to the next copyright extension or the end of the sequence. The copyright_identifier data element is data used for identifying an institution cataloging the copyright specified by the ISO/IEC JTC/SC29. The original_or_copy data element is a flag used for indicating whether data of the bitstream is original or copied data. The copyright_number_1 data element indicates bits 44 to 63 of a copyright number. The copyright_number_2 data element indicates bits 22 to 43 of the copyright number. The copyright_number_3 data element indicates bits 0 to 21 of the copyright number.

[0320] The data elements defined by the picture_display_extension() function are extension_start_code_identifier, frame_center_horizontal_offset and frame_center_vertical_offset as shown in Fig. 58.

[0321] The data elements listed above are described

as follows. The extension_start_code_identifier data element is a code used for indicating which extension data is transmitted. The frame_center_horizontal_offset data element is an offset of the display area in the horizontal direction. The number of such horizontal offsets can be defined by number_of_frame_center_offsets. The frame_center_vertical_offset data element is an offset of the display area in the vertical direction. The number of such vertical offsets can be defined by number_of_frame_center_offsets.

[0322] As shown in the variable-length history stream of Fig.47, data elements defined by a picture_data() function are described as a history stream after the data elements defined by the extension_and_user(2) function.

[0323] As shown in Fig. 59, the data elements defined by a picture_data() function are data elements defined by a slice() function. It should be noted that the data elements defined by a slice() function are not described in the bitstream if slice_start_code representing the start code of the slice() function does not exist in the bitstream.

[0324] As shown in Fig. 60, the slice() function is a function used for describing data elements such as slice_start_code, slice_quantiser_scale_code, intra_slice_flag, intra_slice, reserved_bits, extra_bit_slice, extra_information_slice and extra_bit_slice and data elements defined by a macroblock() function as a history stream.

[0325] The data elements listed above are described as follows. The slice_start_code data element is a start code used for indicating the data elements defined by the slice() function. The slice_quantiser_scale_code data element is the size of the quantization step defined for a macroblock existing in the slice layer. However, quantiser_scale_code set for macroblocks is preferably used, when quantiser_scale_code has been set.

[0326] The intra_slice_flag data segment is a flag used for indicating whether or not intra_slice and reserved_bits exist in the bit stream. The intra_slice data element is a flag used for indicating whether or not a non-intra-macroblock exists in the slice layer. To be more specific, if any one of macroblocks in the slice layer is a non-intra-macroblock, the intra_slice flag has a value of "0" If all macroblocks in the slice layer are non-intra-macroblocks, on the other hand, the intra_slice flag has a value of "1". The reserved_bits data element is 7-bit data having a value of "0". The extra_bit_slice data element is a flag used for indicating whether or not the extra_information_slice data element, that is, information added as a history stream, exists. To be more specific, if the next extra_information_slice data element exists, the extra_bit_slice flag has a value of "1". If the next extra_information_slice data element does not exist, on the other hand, the extra_bit_slice flag has a value of "0".

[0327] Following to the data element defined by the slice() function, data elements defined by a macroblock() function are described as a history stream.

[0328] As shown in Fig. 61, the macroblock() function are a function used for defining data elements such as macroblock_escape, macroblock_address_increment and macroblock_quantiser_scale_code and data elements defined by a macroblock_modes() function and a macroblock_vectors(s) function.

[0329] The data elements listed above are described as follows. The macroblock_escape data element is a string of bits with a fixed length used for indicating whether or not a difference in the horizontal direction between a referenced macroblock and a preceding macroblock is at least 34 or greater. If the difference in the horizontal direction between a referenced macroblock and a preceding macroblock is at least 34 or greater, 33 is added to the value of the macroblock_address_increment data element. The macroblock_address_increment data element is the difference in the horizontal direction between a referenced macroblock and a preceding macroblock. If one macroblock_escape data element exists before the macroblock_address_increment data element, a value obtained as a result of the addition of 33 to the value of the macroblock_address_increment data element represents the actual difference in the horizontal direction between a referenced macroblock and a preceding macroblock.

[0330] The macroblock_quantiser_scale_code data element is the size of the quantization step set in each macroblock. The slice_quantiser_scale_code data element representing the size of the quantization step of a slice layer is also set in each slice layer. However, macroblock_scale_code set for a macroblock takes precedence of slice_quantiser_scale_code.

[0331] Following to the macroblock_address_increment data element, data elements defined by the macroblock_modes() function are described. As shown in Fig. 62, the macroblock_modes() function is a function used for describing data elements such as macroblock_type, frame_motion_type, field_motion_type and dct_type as a history stream.

[0332] The data elements listed above are described as follows. The macroblock_type data element is data representing the encoding type of the macroblock. To put it concretely, the macroblock_type data element is data with a variable length generated from flags such as macroblock_quant, dct_type_flag, macroblock_motion_forward and macroblock_motion_backward as shown in Figs. 65 to 67. The macroblock_quant flag is a flag used for indicating whether or not macroblock_quantiser_scale_code for setting the size of the quantization step for the macroblock is set. If macroblock_quantiser_scale_code exists in the bitstream, the macroblock_quant flag has a value of "1".

[0333] The dct_type_flag is a flag used for indicating

whether or not dct_type showing that the referenced macroblock has been encoded in the frame-DCT mode or the field-DCT mode exists. In other words, dct_type_flag is a flag used for indicating whether or not the referenced macroblock experienced DCT. If dct_type exists in the bitstream, dct_type_flag has a value of "1". The macroblock_motion_forward is a flag showing whether or not the referenced macroblock has undergone forward prediction. If the referenced macroblock has undergone forward prediction, the macroblock_motion_forward flag has a value of "1". On the other hand, macroblock_motion_backward is a flag showing whether or not the referenced macroblock has undergone backward prediction. If the referenced macroblock has undergone backward prediction, the macroblock_motion_backward flag has a value of "1".

[0334] If the macroblock_motion_forward flag or the macroblock_motion_backward flag has a value of "1", the picture is transferred in the frame-prediction mode and frame_period_frame_dct has a value of "0", a data element representing frame_motion_type is described after a data element representing macroblock_type. It should be noted that frame_period_frame_dct is a flag used for indicating whether or not frame_motion_type exists in the bit stream.

[0335] The frame_motion_type data element is a 2-bit code showing the prediction type of the macroblock of the frame. A frame_motion_type value of "00" indicates that there are two prediction vectors and the prediction type is a field-based prediction type. A frame_motion_type value of "01" indicates that there is one prediction vector and the prediction type is a field-based prediction type. A frame_motion_type value of "10" indicates that there is one prediction vector and the prediction type is a frame-based prediction type. A frame_motion_type value of "11" indicates that there is one prediction vector and the prediction type is a dual-prime prediction type.

[0336] If the macroblock_motion_forward flag or the macroblock_motion_backward flag has a value of "1" and the picture is transferred not in the frame prediction mode, a data element representing field_motion_type is described after a data element representing macroblock_type.

[0337] The field_motion_type data element is a 2-bit code showing the motion prediction of the macroblock of a field. A field_motion_type value of "01" indicates that there is one prediction vector and the prediction type is a field-based prediction type. A field_motion_type value of "10" indicates that there is two prediction vectors and the prediction type is a 18 × 8 macroblock-based prediction type. A field_motion_type value of "11" indicates that there is one prediction vector and the prediction type is a dual-prime prediction type.

[0338] If the picture is transferred in the frame prediction mode, frame_period_frame_dct indicates that frame_motion_type exists in the bitstream and frame_period_frame_dct also indicates that dct_type exists in the bitstream, a data element representing dct_type is described after a data element representing macroblock_type. It should be noted that the dct_type data element is data used for indicating whether the DCT is carried out in the frame-DCT mode or the field-DCT mode.

[0339] As shown in Fig. 61, if the referenced macroblock is either a forward-prediction macroblock or an intra-macroblock completing conceal processing, a data element defined by a motion_vectors(0) function is described. If the referenced macroblock is a backward-prediction macroblock, a data element defined by a motion_vectors(1) function is described. It should be noted that the motion_vectors(0) function is a function used for describing a data element related to a first motion vector and the motion_vectors(1) function is a function used for describing a data element related to a second motion vector.

[0340] As shown in Fig. 63, the motion_vectors(s) function is a function used for describing a data element related to a motion vector.

[0341] If there is one motion vector and the dual-prime prediction mode is not used, data elements defined by motion_vertical_field_select[0][s] and motion_vector[0][s] are described.

[0342] The motion_vertical_field_select[r][s] is a flag used for indicating that the first vector, be it a forward-prediction or backward-prediction vector, is a vector made by referencing the bottom field or the top field. The subscript [r] indicates the first or second vector whereas the subscript [s] indicates a forward-prediction or backward-prediction vector.

[0343] As shown in Fig. 64, the motion_vector(r, s) function is a function used for describing a data array related to motion_code[r][s][t], a data array related to motion_residual[r][s][t] and data representing dmvector[t].

[0344] The motion_code[r][s][t] is data with a variable length used for representing the magnitude of a motion vector in terms of a value in the range -16 to +16. motion_residual[r][s][t] is data with a variable length used for representing a residual of a motion vector. Thus, by using the values of motion_code[r][s][t] and motion_residual[r][s][t], a detailed motion vector can be described. The dmvector[t] is data used for scaling an existing motion vector with a time distance in order to generate a motion vector in one of the top and bottom fields (for example, the top field) in the dual-prime prediction mode and used for correction in the vertical direction in order to reflect a shift in the vertical direction between lines of the top and bottom fields. The subscript [r] indicates the first or second vector whereas the subscript [s] indicates a forward-prediction or backward-prediction vector. The subscript [t] indicates that the motion vector is a component in the vertical or horizontal direction.

[0345] First of all, the motion_vector(r, s) function

describes a data array to represent motion_code[r][s][0] in the horizontal direction as a history stream as shown in Fig. 64. The number of bits of both motion_residual[0][s][t] and motion_residual[1][s][t] is represented by f_code[s][t]. Thus, a value of f_code[s][t] other than "1" indicates that motion_residual[r][s][t] exists in the bitstream. The fact that motion_residual[r][s][0], a horizontal-direction component, is not "1" and motion_code[r][s][0], a horizontal-direction component, is not "0" indicates that a data element representing motion_residual[r][s][0] is included in the bit stream and the horizontal-direction component of the motion vector exists. In this case, a data element representing motion_residual[r][s][0], a horizontal component, is thus described.

[0346] Subsequently, a data array to represent motion_code[r][s][1] is described in the vertical direction as a history stream. Likewise, the number of bits of both motion_residual[0][s][t] and motion_residual[1][s][t] is represented by f_code[s][t]. Thus, a value of f_code[s][t] other than "1" indicates that motion_residual[r][s][t] exists in the bitstream. The fact that motion_residual[r][s][1], a vertical-direction component, is not "1" and motion_code[r][s][1], a vertical-direction component, is not "0" indicates that a data element representing motion_residual[r][s][1] is included in the bitstream and the vertical-direction component of the motion vector exists. In this case, a data element representing motion_residual[r][s][1], a vertical component, is thus described.

[0347] It should be noted that, in the variable-length format, the history information can be eliminated in order to reduce the transfer rate of transmitted bits.

[0348] For example, in order to transfer macroblock_type and motion_vectors(), but not to transfer quantiser_scale_code, slice_quantiser_scale_code is set at "00000" in order to reduce the bit rate.

[0349] In addition, in order to transfer only macroblock_type but not to transfer motion_vectors(), quantiser_scale_code and dct_type, "not_coded" is used as macroblock_type in order to reduce the bit rate.

[0350] Furthermore, in order to transfer only picture_coding_type but not to transfer all information following slice(), picture_data() having no slice_start_code is used in order to reduce the bit rate.

[0351] As described above, in order to prevent 23 consecutive bits of 0 from appearing in user_data, a "1" bit is inserted for every 22 bits. It should be noted that, however, a "1" bit can also be inserted for each number of bits smaller than 22. In addition, instead of insertion of a "1" bit by counting the number of consecutive 0 bits, a "1" bit can also be inserted by examining Byte_allign.

[0352] In addition, in the MPEG, generation of 23 consecutive bits of 0 is prohibited. In actuality, however, only a sequence of such 23 bits starting from the beginning of a byte is a problem. That is to say, a sequence of such 23 bits not starting from the beginning of a byte is not a problem. Thus, a "1" bit may be inserted for each typically 24 bits at a position other than the LSB.

[0353] Furthermore, while the history information is made in a format close to a video elementary stream as described above, the history information can also be made in a format close to a packetized elementary stream or a transport stream. In addition, even though user_data of the Elementary Stream is placed in front of picture_data according to the above description, user_data can be placed at another location as well.

[0354] It should be noted that programs to be executed by a computer for carrying out pieces of processing described above can be presented to the user through network presentation media such as the Internet or a digital satellite in addition to presentation media implemented by an information recording medium such as a magnetic disc and a CD-ROM.

Claims

1. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, the system comprising:

   decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting past encoding parameters generated by past encoding processes from said source encoded video stream;
   encoding means for encoding said video data as a present encoding process to generate an encoded video stream, and
   control means for receiving said past encoding parameters generated, and for controlling said present encoding process of said encoding means based on said past encoding parameters.

2. The transcoding system according to claim 1, wherein

   said encoding means encodes a reference picture included in said source video data with a present picture type assigned to said reference picture in said present encoding process; and
   said control means judges whether said reference picture has been encoded with the same picture type as said assigned picture type in the past encoding processes, and controls said present encoding process based on a consequence of the judgment.

3. The transcoding system according to claim 1, wherein

   said encoding means encodes a reference picture included in said video data with a present picture type assigned to said reference picture in said present encoding process, and

said control means detects past picture types which have been assigned to said reference picture at said present encoding processes by referring said past encoding parameters, and then, said control means controls said present encoding process based on said present picture type and said past picture types.

4. The transcoding system according to claim 2, wherein

said control means selects optimum encoding parameters from said past encoding parameters according to said judgement, and controls said present encoding process of said encoding means based on said selected optimum encoding parameters.

5. The transcoding system according to claim 2, wherein

said control means encodes said reference picture by using said past encoding parameters generated at one of said past encoding processes.

6. The transcoding system according to claim 4, wherein

said past encoding parameters include motion vector information generated at said past encoding processes; and
said encoding means includes a motion vector detect means for detecting a motion vector information of said reference picture in said present encoding processes.

7. The transcoding system according to claim 6, wherein

said control means controls an operation of said motion vector detect means said based on said result of the judgment.

8. The transcoding system according to claim 7, wherein

said control means reuses said motion vector information included in said past encoding parameters as substitute for calculation of new motion vector information in said motion vector detect means.

9. The transcoding system according to claim 7, wherein

said control means reuses said motion vector information included in said past encoding processes, if said reference picture has been encoded with the same picture type as said present picture type in past encoding processes.

10. The transcoding system according to claim 9, wherein

said control means controls said motion vector detect means so that new motion vector information was detected by said motion vector detect means detects, if said reference picture has not been encoded with said assigned picture type at past encoding process.

11. The transcoding system according to claim 1, wherein

said control means selects optimum encoding parameters which are commensurate with said present encoding process from said past encoding parameters, and controls said present encoding process of said encoding means based on said optimum encoding parameters.

12. The transcoding system according to claim 1, wherein

said encoding means encodes a reference picture included in said source video data with a present picture type assigned to said reference picture in said present encoding process; and
said control means judges the fact of whether said reference picture has been encoded with the same picture type as said assigned picture type at the past encoding processes, and controls selects said optimum encoding parameters based on a consequence of the judgment.

13. The transcoding system according to claim 11, wherein

said past encoding parameters includes a prediction mode information indicating a flame prediction mode or a field prediction mode; and
said control means controls said present encoding process according to said prediction mode information.

14. The transcoding system according to claim 11, wherein

said control means reuses said prediction mode information included in said past encoding parameters as substitute for calculation of new prediction mode information, if said reference picture has been encoded with the same

picture type as said present picture type at past encoding processes.

15. The transcoding system according to claim 11, wherein

said encoding parameters includes a prediction type information indicating intra prediction, forward prediction, backward prediction, or interpolative prediction; and
said control means controls said present encoding process based on said prediction type information.

16. The transcoding system according to claim 15, wherein

said control means reuses said prediction type information included in said past encoding parameters as substitute for calculation of new prediction type information, if said reference picture has been encoded with the same picture type as said present picture type at past encoding processes.

17. The transcoding system according to claim 11, wherein

said encoding parameters includes a DCT mode information indicating a flame DCT mode or a field DCT mode; and
said control means controls said present encoding process based on said DCT mode information.

18. The transcoding system according to claim 17, wherein

said control means reuses said DCT mode information included in said past encoding parameters as a substitute for calculation of new DCT mode information, if said reference picture has been encoded with the same picture type as said present picture type at past encoding processes.

19. The transcoding system according to claim 1, wherein

said control means generates present encoding parameters corresponds to said present encoding process of said encoding means.

20. The transcoding system according to claim 19, wherein

said control means selects optimum encoding parameter which are commensurate with said present encoding process from said present encoding parameter and said past encoding parameters, and controls said present encoding process of said encoding means based on said optimum encoding parameter.

21. The transcoding system according to claim 20, wherein

said encoding means encodes a reference picture included in said source video data with a present picture type assigned to said reference picture in said present encoding process; and
said control means judges the fact of whether said reference picture has been encoded with the same picture type as said assigned picture type at the past encoding processes, and controls selects said optimum encoding parameter based on a consequence of the judgment.

22. The transcoding system according to claim 21, wherein

said past encoding parameters include a quantization information generated at said past encoding processes; and
said encoding means includes a quatization means for quantizating said reference picture in said present encoding processes.

23. The transcoding system according to claim 22, wherein

said control means receives a buffer information indicating a fullness of a transmission buffer for storing said encoded video stream, and controls said quantization means based on said buffer information so as to prevent overflow and underflow of said transmission buffer.

24. The transcoding system according to claim 23, wherein

said control means controls said quantization means based on a quantization step size derived from said buffer information and quantization step sizes derived from said quantization information included in said past encoding parameter.

25. The transcoding system according to claim 24, wherein

said control means controls said quantization means by using the biggest quantization step size selected from among said quantization step size corresponds to said buffer information and said quantization step sizes correspond to

said quantization information.

26. The transcoding system according to claim 1, wherein

said control means controls said encoding means so that said encoding means describes said past encoding parameters into said encoded video stream.

27. The transcoding system according to claim 1, wherein

said encoding means includes a processing means for processing said encoded video stream to generate MPEG bit stream in conformance with MPEG standards, said MPEG bit stream comprising sequence layer, GOP layer, picture layer, slice layer, and macroblock layer.

28. The transcoding system according to claim 27, wherein:

said control means generates present encoding parameters corresponds to said present encoding process of said encoding means; and said processing means describes said present encoding parameters into said picture layer, said slice layer, and said macroblock layer, and describes said past encoding parameters into a user data area provided in said picture layer.

29. The transcoding system according to claim 28, wherein

said processing means generates a history stream comprising said past encoding parameters in order to describe said past encoding parameters in said user data area.

30. The transcoding system according to claim 29, wherein

said processing means inserts marker bits into said history stream in order to prevent a emulation of fixed start code defined with MPEG standards, describes said history stream inserted said marker bits in said user data area provided in said picture layer.

31. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding the source encoded video stream to generate decoded video data;
extracting past encoding parameters generated by past encoding processes from the source encoded video stream;
encoding the video data as a present encoding process to generate an encoded video stream;
receiving the past encoding parameters generated; and
controlling the present encoding process of the encoding means based on the past encoding parameters.

32. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, for extracting past encoding parameters generated by past encoding processes from said source encoded video stream, and for outputting said past encoding parameters as history information;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process; and
control means for receiving said history information including past encoding parameters, and for controlling said present encoding process of said encoding means based on said history information so that said present encoding process being optimized by selectively using said past encoding parameters.

33. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding the source encoded video stream to generate decoded video data;
extracting past encoding parameters generated by past encoding processes from said source encoded video stream;
outputting said past encoding parameters as history information;
encoding said decoded video data to generate an encoded video stream as a present encoding process;
receiving said history information including past encoding parameters; and
controlling said present encoding process of said encoding means based on said history information so that said present encoding process being optimized by selectively using said past encoding parameters

34. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process; and
control means for receiving said past encoding parameters, for selecting optimum encoding parameters which are commensurate with said present encoding process from said past encoding parameters, and for controlling said present encoding process of said encoding means based on said optimum encoding parameters.

35. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding said decoded video data to generate an encoded video stream as a present encoding process;
receiving said past encoding parameters;
selecting optimum encoding parameters which are commensurate with said present encoding process from said past encoding parameters; and
controlling said present encoding process of said encoding means based on said optimum encoding parameters.

36. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding a reference picture included in said encoded video data with a assigned picture type; and
control means for receiving past encoding parameters generated at past encoding processes, for selecting optimum encoding parameters from said past encoding parameters according to said assigned picture type, and for controlling a present encoding process of said encoding means based on said optimum encoding parameters.

37. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding a reference picture included in said encoded video data with a assigned picture type;
receiving past encoding parameters generated at past encoding processes;
selecting optimum encoding parameters from said past encoding parameters according to said assigned picture type; and
controlling a present encoding process of said encoding means based on said optimum encoding parameters.

38. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding a reference picture included in said decoded video data with a assigned picture type; and
control means for judging whether the reference picture has been encoded with said assigned picture type at the past encoding process, and for controlling a present encoding process of said encoding means based on a consequence of the judgment.

39. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding a reference picture included in said decoded video data with a assigned picture type;
judging whether the reference picture has been encoded with said assigned picture type at the past encoding process; and
controlling a present encoding process of said encoding means based on a consequence of the judgment

40. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process; and
stream generating means for generating a MPEG stream including said encoded video data, present encoding parameters generated at said present encoding process, and said past encoding parameters generated at past encoding processes.

41. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process; and
generating a MPEG stream including said encoded video data, present encoding parameters generated at said present encoding process, and said past encoding parameters generated at past encoding processes.

42. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process; and
stream generating means for generating a MPEG bit stream comprising of sequence layer, GOP layer, picture layer, slice layer, and macroblock layer, wherein each layer include present encoding parameters generated at said present encoding process, and wherein said picture layer also includes said past encoding parameters generated at past encoding processes.

43. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process; and
generating a MPEG bit stream comprising of sequence layer, GOP layer, picture layer, slice layer, and macroblock layer, wherein each layer include present encoding parameters generated at said present encoding process, and wherein said picture layer also includes said past encoding parameters generated at past encoding processes.

44. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process by referring said past encoding parameters;
describe means for describing said past encoding parameters into said encoded video stream; and
output means for outputting said encoded video stream in which said past parameters being described.

45. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding said decoded video data to generate an encoded video stream as a present encoding process by referring said past encoding parameters;
describing said past encoding parameters into said encoded video stream; and
outputting said encoded video stream in which said past parameters being described

46. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, the system comprising:

means for decoding said source encoded video stream to generate decoded video data;
means for extracting a history information including a previous encoding parameters generated by past encoding processes from said source encoded video stream;
means for encoding said decoded video data to generate an encoded video stream as a present encoding process by referring said history information; and
means for describing said history information in said encoded stream so that said history information is available in advance encoding process.

47. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a history information including a previous encoding parameters generated by past encoding processes from said source encoded video stream;
encoding said decoded video data to generate an encoded video stream as a present encoding process by referring said history information; and
describing said history information in said encoded stream so that said history information is available in advance encoding process.

48. A transcoding system for converting a bit rate or a GOP structure of source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream based on encoding parameters of latest encoding process to generate base-band video data, and for extracting past encoding parameters of past encoding processes, and for multiplexing said encoding parameters of said latest and past encoding process into said base-band video data; and
encoding means for encoding said base-band video data based on said encoding parameters of said least and past encoding process to generate a new encoded video stream so that bit rate or GOP structure of said new encoded video stream are different from said bit rate or GOP structure of said source encoded video stream.

49. A method for converting a bit rate or a GOP structure of source encoded video stream, comprising the steps of:

decoding said source encoded video stream based on encoding parameters of latest encoding process to generate base-band video data;
extracting past encoding parameters of past encoding processes;
multiplexing said encoding parameters of said latest and past encoding process into said base-band video data; and
encoding said base-band video data based on said encoding parameters of said least and past encoding process to generate a new encoded video stream so that bit rate or GOP structure of said new encoded video stream are different from said bit rate or GOP structure of said source encoded video stream.

50. A video encoding apparatus for encoding a source video data, the apparatus comprising:

encoding means for encoding said video data as a present encoding process to generate an encoded video stream; and
control means for receiving past encoding parameters generated by past encoding processes, and for controlling said present encoding process of said encoding means based on said past encoding parameters.

51. The video encoding apparatus according to claim 50, wherein

said encoding means encodes a reference picture included in said source video data with a present picture type assigned to said reference picture in said present encoding process; and
said control means judges the fact of whether said reference picture has been encoded with the same picture type as said assigned picture type at the past encoding processes, and controls said present encoding process based on a consequence of the judgment.

52. The video encoding apparatus according to claim 50, wherein

said encoding means encodes a reference picture included in said video data with a present picture type assigned to said reference picture in said present encoding process; and
said control means detects past picture types which have been assigned to said reference picture at said present encoding processes by referring said past encoding parameters, and then, said control means controls said present

encoding process based on said present picture type and said past picture types.

53. The video encoding apparatus according to claim 51, wherein

said control means selects optimum encoding parameters from said past encoding parameters according to said judgement, and controls said present encoding process of said encoding means based on said selected optimum encoding parameters.

54. The video encoding apparatus according to claim 51, wherein

said control means encodes said reference picture by using said past encoding parameters generated at one of said past encoding processes.

55. The video encoding apparatus according to claim 53, wherein

said past encoding parameters include motion vector information generated at said past encoding processes; and
said encoding means includes a motion vector detect means for detecting a motion vector information of said reference picture in said present encoding processes.

56. The video encoding apparatus according to claim 55, wherein

said control means controls an operation of said motion vector detect means said based on said result of the judgment.

57. The video encoding apparatus according to claim 56, wherein

said control means reuses said motion vector information included in said past encoding parameters as substitute for calculation of new motion vector information in said motion vector detect means.

58. The video encoding apparatus according to claim 56, wherein

said control means reuses said motion vector information included in said past encoding processes, if said reference picture has been encoded with the same picture type as said present picture type at past encoding processes.

59. The video encoding apparatus according to claim 58, wherein

said control means controls said motion vector detect means so that new motion vector information was detected by said motion vector detect means detects, if said reference picture has not been encoded with said assigned picture type at past encoding process.

60. The video encoding apparatus according to claim 50, wherein

said control means selects optimum encoding parameters which are commensurate with said present encoding process from said past encoding parameters, and controls said present encoding process of said encoding means based on said optimum encoding parameters.

61. The video encoding apparatus according to claim 50, wherein

said encoding means encodes a reference picture included in said source video data with a present picture type assigned to said reference picture in said present encoding process; and
said control means judges the fact of whether said reference picture has been encoded with the same picture type as said assigned picture type at the past encoding processes, and controls selects said optimum encoding parameters based on a consequence of the judgment.

62. The video encoding apparatus according to claim 60, wherein

said past encoding parameters includes a prediction mode information indicating a flame prediction mode or a field prediction mode; and
said control means controls said present encoding process according to said prediction mode information.

63. The video encoding apparatus according to claim 62, wherein

said control means reuses said prediction mode information included in said past encoding parameters as substitute for calculation of new prediction mode information, if said reference picture has been encoded with the same picture type as said present picture type at past encoding processes.

64. The video encoding apparatus according to claim 50, wherein

said encoding parameters includes a prediction type information indicating intra prediction, forward prediction, backward prediction, or interpolative prediction;

said control means controls said present encoding process based on said prediction type information.

65. The video encoding apparatus according to claim 64, wherein

said control means reuses said prediction type information included in said past encoding parameters as substitute for calculation of new prediction type information, if said reference picture has been encoded with the same picture type as said present picture type at past encoding processes.

66. The video encoding apparatus according to claim 60, wherein

said encoding parameters includes a DCT mode information indicating a flame DCT mode or a field DCT mode;

said control means controls said present encoding process based on said DCT mode information.

67. The video encoding apparatus according to claim 66, wherein said control means reuses said DCT mode information included in said past encoding parameters as substitute for calculation of new DCT mode information, if said reference picture has been encoded with the same picture type as said present picture type at past encoding processes.

68. The video encoding apparatus according to claim 50, wherein

said control means generates present encoding parameters corresponds to said present encoding process of said encoding means.

69. The video encoding apparatus according to claim 68, wherein

said control means selects optimum encoding parameter which are commensurate with said present encoding process from said present encoding parameter and said past encoding parameters, and controls said present encoding process of said encoding means based on said optimum encoding parameter.

70. The video encoding apparatus according to claim 69, wherein

said encoding means encodes a reference picture included in said source video data with a present picture type assigned to said reference picture in said present encoding process; and

said control means judges the fact of whether said reference picture has been encoded with the same picture type as said assigned picture type at the past encoding processes, and controls selects said optimum encoding parameter based on a consequence of the judgment.

71. The video encoding apparatus according to claim 70, wherein

said past encoding parameters include a quantization information generated at said past encoding processes; and

said encoding means includes a quantization means for quantizating said reference picture in said present encoding processes.

72. The video encoding apparatus according to claim 71, wherein

said control means receives a buffer information indicating a fullness of a transmission buffer for storing said encoded video stream, and controls said quantization means based on said buffer information so as to prevent overflow and underflow of said transmission buffer.

73. The video encoding apparatus according to claim 72, wherein

said control means controls said quantization means based on a quantization step size derived from said buffer information and quantization step sizes derived from said quantization information included in said past encoding parameter.

74. The video encoding apparatus according to claim 73, wherein

said control means controls said quantization means by using the biggest quantization step size selected from among said quantization step size corresponds to said buffer information and said quantization step sizes correspond to said quantization information.

75. The video encoding apparatus according to claim 50, wherein

said control means controls said encoding means so that said encoding means describes said past encoding parameters into said encoded video stream.

76. The video encoding apparatus according to claim 50, wherein

said encoding means includes a processing means for processing said encoded video stream to generate MPEG bit stream in conformance with MPEG standards, said MPEG bit stream comprising sequence layer, GOP layer, picture layer, slice layer, and macroblock layer.

77. The video encoding apparatus according to claim 76, wherein;

said control means generates present encoding parameters corresponds to said present encoding process of said encoding means;
said processing means describes said present encoding parameters into said picture layer, said slice layer, and said macroblock layer, and describes said past encoding parameters into a user data area provided in said picture layer.

78. The video encoding apparatus according to claim 77, wherein

said processing means generates a history stream comprising said past encoding parameters in order to describe said past encoding parameters in said user data area.

79. The video encoding apparatus according to claim 78, wherein

said processing means inserts marker bits into said history stream in order to prevent a emulation of fixed start code defined with MPEG standards, describes said history stream inserted said marker bits in said user data area provided in said picture layer.

80. A video encoding apparatus for encoding a source video data, the apparatus comprising:

encoding means for encoding said video data to generate said encoded video stream as a present encoding process; and
control means for receiving history information including past encoding parameters generated at past encoding processes, and for controlling a present encoding process of said encoding means based on said history information so that said present encoding process being optimized by selectively using said past encoding parameters.

81. A video encoding method for encoding a source video data, comprising the steps of:

encoding said video data to generate said encoded video stream as a present encoding process; and
receiving history information including past encoding parameters generated at past encoding processes, and for controlling a present encoding process of said encoding means based on said history information so that said present encoding process being optimized by selectively using said past encoding parameters.

82. A video encoding apparatus for encoding a video data to generate encoded video stream, the apparatus comprising:

encoding means for encoding said video data to generate said encoded video stream as a present encoding process; and
control means for receiving past encoding parameters generated at past encoding processes, for selecting optimum encoding parameters which are commensurate with said present encoding process from said past encoding parameters, and for controlling a present encoding process of said encoding means based on said optimum encoding parameters.

83. A video encoding method for encoding a video data to generate encoded video stream, comprising the steps of:

encoding said video data to generate said encoded video stream as a present encoding process; and
receiving past encoding parameters generated at past encoding processes, for selecting optimum encoding parameters which are commensurate with said present encoding process from said past encoding parameters, and for controlling a present encoding process of said encoding means based on said optimum encoding parameters.

84. A video encoding apparatus for encoding a video data to generate encoded video stream, the apparatus comprising:

encoding means for encoding a reference picture included in said video data with a assigned picture type; and
control means for receiving past encoding parameters generated at past encoding processes, for selecting optimum encoding parameters from said past encoding parameters according to said assigned picture type, and for controlling a present encoding process of said

encoding means based on said optimum encoding parameters.

85. A video encoding method for encoding a video data to generate encoded video stream, comprising the steps of:

encoding a reference picture included in said video data with a assigned picture type;
receiving past encoding parameters generated at past encoding processes, for selecting optimum encoding parameters from said past encoding parameters according to said assigned picture type; and
controlling a present encoding process of said encoding means based on said optimum encoding parameters.

86. A video encoding apparatus for encoding a video data to generate encoded video stream, the apparatus comprising:

encoding means for encoding a reference picture included in said video data with a assigned picture type; and
control means for judging whether the reference picture has been encoded with said assigned picture type at the past encoding process, and for controlling a present encoding process of said encoding means based on a consequence of the judgment.

87. A video encoding method for encoding a video data to generate encoded video stream, comprising the steps of:

encoding a reference picture included in said video data with a assigned picture type;
judging whether the reference picture has been encoded with said assigned picture type at the past encoding process; and
controlling a present encoding process of said encoding means based on a consequence of the judgment.

88. A video encoding apparatus for encoding a source video data, the apparatus comprising:

encoding means for said video data to generate encoded video data as a present encoding process; and
stream generating means for generating a MPEG bit stream including said encoded video data, present encoding parameters generated at said present encoding process, and past encoding parameters generated at past encoding process.

89. A video encoding method for encoding a source video data, comprising the steps of:

generate encoded video data as a present encoding process; and
generating a MPEG bit stream including said encoded video data, present encoding parameters generated at said present encoding process, and past encoding parameters generated at past encoding process.

90. A video encoding apparatus for encoding a video data to generate encoded video stream, the apparatus comprising:

encoding means for said video data to generate encoded video as present encoding process; and
stream generating means for generating a MPEG bit stream comprising of sequence layer, GOP layer, picture layer, slice layer, and macroblock layer, wherein each layer include present encoding parameters generated at said present encoding process, and wherein said picture layer also includes past encoding parameters generated at past encoding process.

91. A video encoding method for encoding a video data to generate encoded video stream, comprising the steps of:

generate encoded video as present encoding process; and
generating a MPEG bit stream comprising of sequence layer, GOP layer, picture layer, slice layer, and macroblock layer, wherein each layer include present encoding parameters generated at said present encoding process, and wherein said picture layer also includes past encoding parameters generated at past encoding process.

92. A video encoding apparatus for encoding a video data to generate encoded video stream, the apparatus comprising:

encoding means for encoding a source video data by referring the past encoding parameters generated at past encoding process to generate said encoded video stream;
describe means for describing said past encoding parameters into said encoded video stream; and
output means for outputting said encoded video stream in which said past parameters being described.

93. A video encoding method for encoding a video data to generate encoded video stream, comprising the steps of:

encoding a source video data by referring the past encoding parameters generated at past encoding process to generate said encoded video stream;
describing said past encoding parameters into said encoded video stream; and
outputting said encoded video stream in which said past parameters being described.

94. A video encoding apparatus for encoding a source video data, the apparatus comprising:

means for receiving a history information including a plurality of encoding parameters generated at previous encoding processes;
encoding means for encoding said source video data by referring said history information; and
means for describing said history information in said encoded stream so that said history information is available in advance encoding process.

95. A video encoding method for encoding a source video data, comprising the steps of:

receiving a history information including a plurality of encoding parameters generated at previous encoding processes;
encoding said source video data by referring said history information; and
describing said history information in said encoded stream so that said history information is available in advance encoding process.

96. A stream processing system for processing a source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding said video data as a present encoding process to generate an encoded video stream; and
control means for receiving said past encoding parameters generated, and for controlling said present encoding process of said encoding means based on said past encoding parameters.

97. A stream processing method for processing a source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding said video data as a present encoding process to generate an encoded video stream; and
receiving said past encoding parameters generated, and for controlling said present encoding process of said encoding means based on said past encoding parameters.

98. A stream processing system for processing a source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream, and for outputting said past encoding parameters as a history information;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process; and
control means for receiving said history information including past encoding parameters, and for controlling said present encoding process of said encoding means based on said history information so that said present encoding process being optimized by selectively using said past encoding parameters.

99. A stream processing method for processing a source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
outputting said past encoding parameters as a history information;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process;
receiving said history information including past encoding parameters; and
controlling said present encoding process of said encoding means based on said history information so that said present encoding process being optimized by selectively using said

past encoding parameters.

100. A stream processing system for processing a source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process; and
control means for receiving said past encoding parameters, for selecting optimum encoding parameters which are commensurate with said present encoding process from said past encoding parameters, and for controlling said present encoding process of said encoding means based on said optimum encoding parameters.

101. A stream processing method for processing a source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding said decoded video data to generate an encoded video stream as a present encoding process;
receiving said past encoding parameters, for selecting optimum encoding parameters which are commensurate with said present encoding process from said past encoding parameters; and
controlling said present encoding process of said encoding means based on said optimum encoding parameters.

102. A stream processing system for processing a source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding a reference picture included in said encoded video data with a assigned picture type; and
control means for receiving past encoding parameters generated at past encoding processes, for selecting optimum encoding parameters from said past encoding parameters according to said assigned picture type, and for controlling a present encoding process of said encoding means based on said optimum encoding parameters.

103. A stream processing method for processing a source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding a reference picture included in said encoded video data with a assigned picture type; receiving past encoding parameters generated at past encoding processes;
selecting optimum encoding parameters from said past encoding parameters according to said assigned picture type; and
controlling a present encoding process of said encoding means based on said optimum encoding parameters.

104. A stream processing system for processing a source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding a reference picture included in said decoded video data with a assigned picture type; and
control means for judging the fact of whether the reference picture has been encoded with said assigned picture type at the past encoding process, and for controlling a present encoding process of said encoding means based on a consequence of the judgment.

105. A stream processing method for processing a source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;
extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;
encoding means for encoding a reference picture included in said decoded video data with a

assigned picture type;

control means for judging the fact of whether the reference picture has been encoded with said assigned picture type at the past encoding process; and

controlling a present encoding process of said encoding means based on a consequence of the judgment.

106. A stream processing system for processing a source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;

encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process;

stream generating means for generating a MPEG stream including said encoded video data, present encoding parameters generated at said present encoding process, and said past encoding parameters generated at past encoding processes.

107. A stream processing method for processing a source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;

extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;

encoding said decoded video data to generate an encoded video stream as a present encoding process; and

stream generating means for generating a MPEG stream including said encoded video data, present encoding parameters generated at said present encoding process, and said past encoding parameters generated at past encoding processes.

108. A stream processing system for processing a source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;

encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process; and

stream generating means for generating a MPEG bit stream comprising a sequence layer, GOP layer, picture layer, slice layer, and macroblock layer, wherein each layer include present encoding parameters generated at said present encoding process, and wherein said picture layer also includes said past encoding parameters generated at past encoding processes.

109. A stream processing method for processing a source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;

extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;

encoding said decoded video data to generate an encoded video stream as a present encoding process;

generating a MPEG bit stream comprising a sequence layer, GOP layer, picture layer, slice layer, and macroblock layer, wherein each layer include present encoding parameters generated at said present encoding process, and wherein said picture layer also includes said past encoding parameters generated at past encoding processes.

110. A stream processing system for processing a source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream to generate decoded video data, and for extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;

encoding means for encoding said decoded video data to generate an encoded video stream as a present encoding process by referring said past encoding parameters;

describe means for describing said past encoding parameters into said encoded video stream; and

output means for outputting said encoded video stream in which said past parameters being described.

111. A stream processing method for processing a source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;

extracting a past encoding parameters generated by past encoding processes from said source encoded video stream;

encoding said decoded video data to generate an encoded video stream as a present encoding process by referring said past encoding parameters;

describing said past encoding parameters into said encoded video stream; and

output means for outputting said encoded video stream in which said past parameters being described.

112. A stream processing system for processing a source encoded video stream, the system comprising:

means for decoding said source encoded video stream to generate decoded video data;

means for extracting a history information including a previous encoding parameters generated by past encoding processes from said source encoded video stream;

means for encoding said decoded video data to generate an encoded video stream as a present encoding process by referring said history information; and

means for describing said history information in said encoded stream so that said history information is available in advance encoding process.

113. A stream processing method for processing a source encoded video stream, comprising the steps of:

decoding said source encoded video stream to generate decoded video data;

extracting a history information including a previous encoding parameters generated by past encoding processes from said source encoded video stream;

encoding said decoded video data to generate an encoded video stream as a present encoding process by referring said history information; and

describing said history information in said encoded stream so that said history information is available in advance encoding process.

114. A stream processing system for processing a source encoded video stream, the system comprising:

decoding means for decoding said source encoded video stream based on encoding parameters of latest encoding process to generate base-band video data, and for extracting past encoding parameters of past encoding processes, and for multiplexing said encoding parameters of said latest and past encoding process into said base-band video data;

encoding means for encoding said base-band video data based on said encoding parameters of said least and past encoding process to generate a new encoded video stream so that bit rate or GOP structure of said new encoded video stream are different from said bit rate or GOP structure of said source encoded video stream.

115. A video decoding method for decoding source encoded video stream, comprising the steps of:

parsing a syntax of said source encoded video stream to extract a past encoding parameters generated at past encoding process;

decoding said encoded video stream; and

outputting said decoded video data and said past encoding parameters so that said past encoding parameters are available in advance encoding process of said decoded video data.

116. A video decoding apparatus for decoding source encoded video stream, the apparatus comprising:

parsing means for parsing a syntax of said source encoded video stream to extract a present encoding parameter generated at least encoding process and a past encoding parameters generated at past encoding process;

decoding means for decoding said encoded video stream based on said present encoding parameter; and

means for outputting said decoded video data, said present encoding parameter and said past encoding parameters so that both of said present and past encoding parameters are available in advance encoding process of said decoded video data.

117. A video decoding method for decoding source encoded video stream, comprising the steps of:

parsing a syntax of said source encoded video stream to extract a present encoding parameter generated at least encoding process and a past encoding parameters generated at past encoding process;

decoding said encoded video stream based on said present encoding parameter; and

outputting said decoded video data, said present encoding parameter and said past encoding parameters so that both of said present and past encoding parameters are available in advance encoding process of said

decoded video data.

FIG. 1

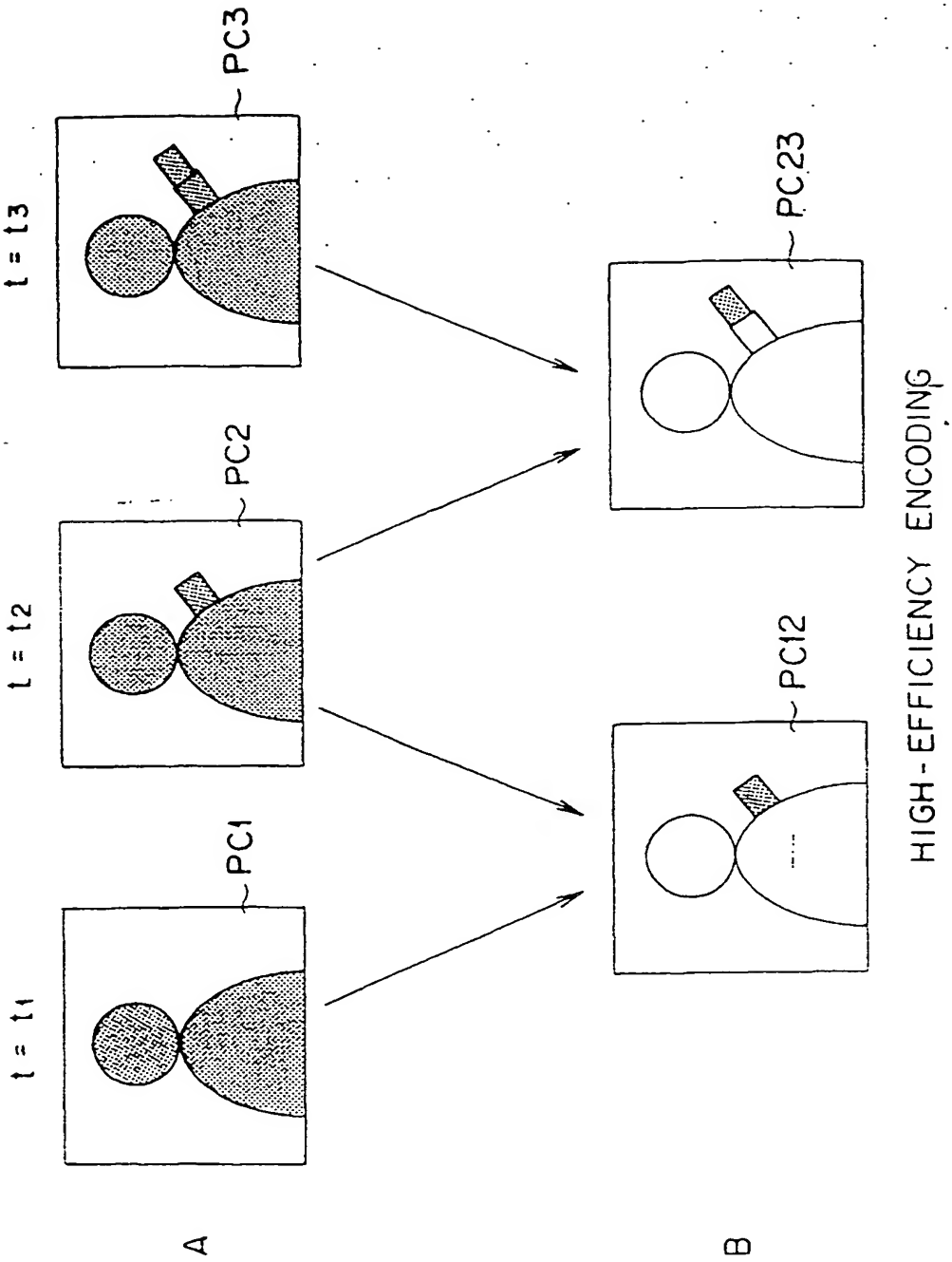
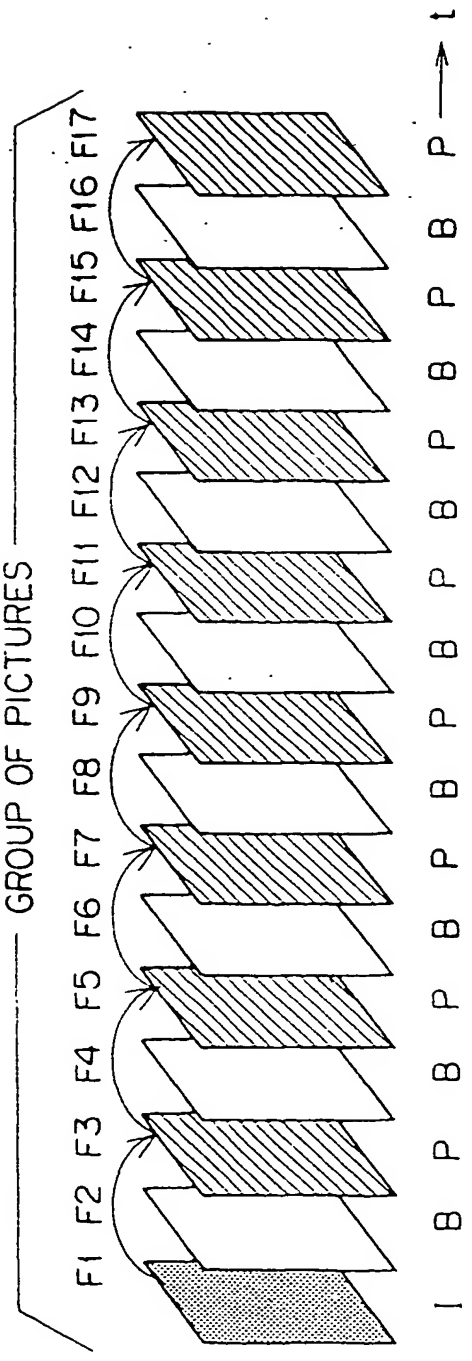

t = t1
PC1
t = t2
PC2
t = t3
PC3
A
PC12
PC23
B
HIGH-EFFICIENCY ENCODING

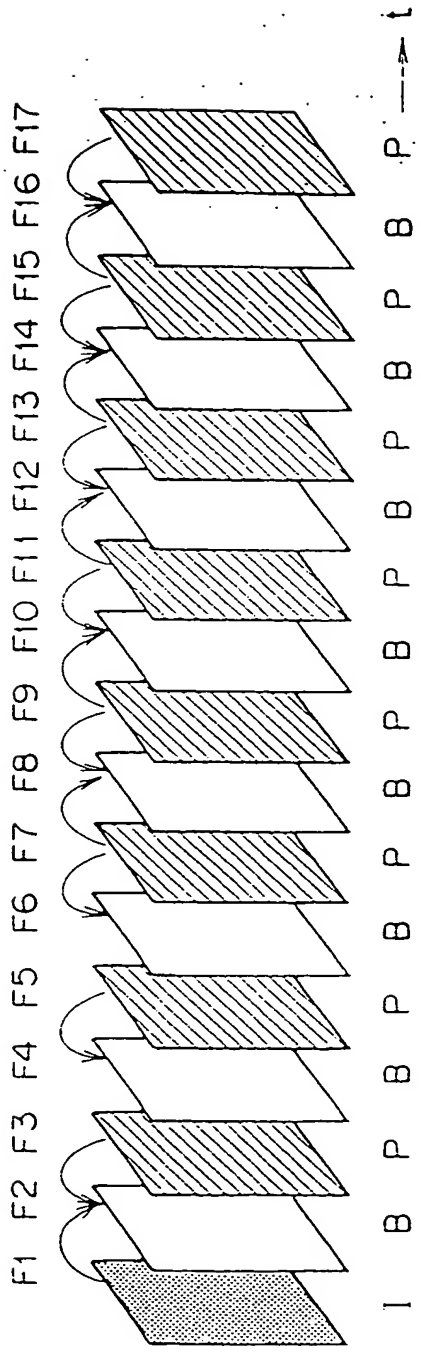
FIG.2



P-PICTURES: SUBJECTED TO FORWARD PREDICTION

PICTURE TYPES: I-, P- AND B-PICTURES

FIG. 3



B-PICTURES: SUBJECTED TO FORWARD & BACKWARD PREDICTION

PICTURE TYPES: I-, P- AND B- PICTURES

FIG. 4

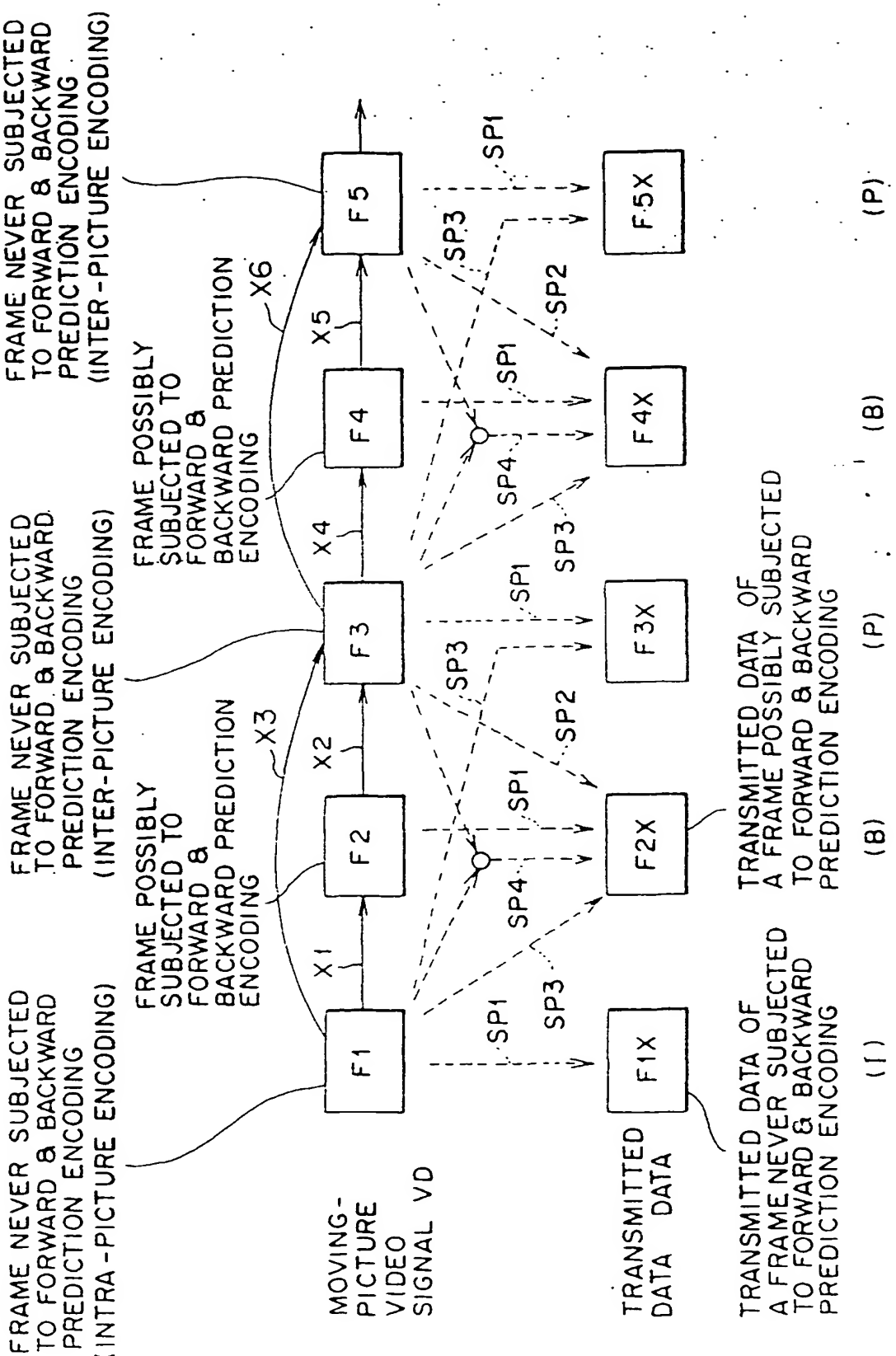
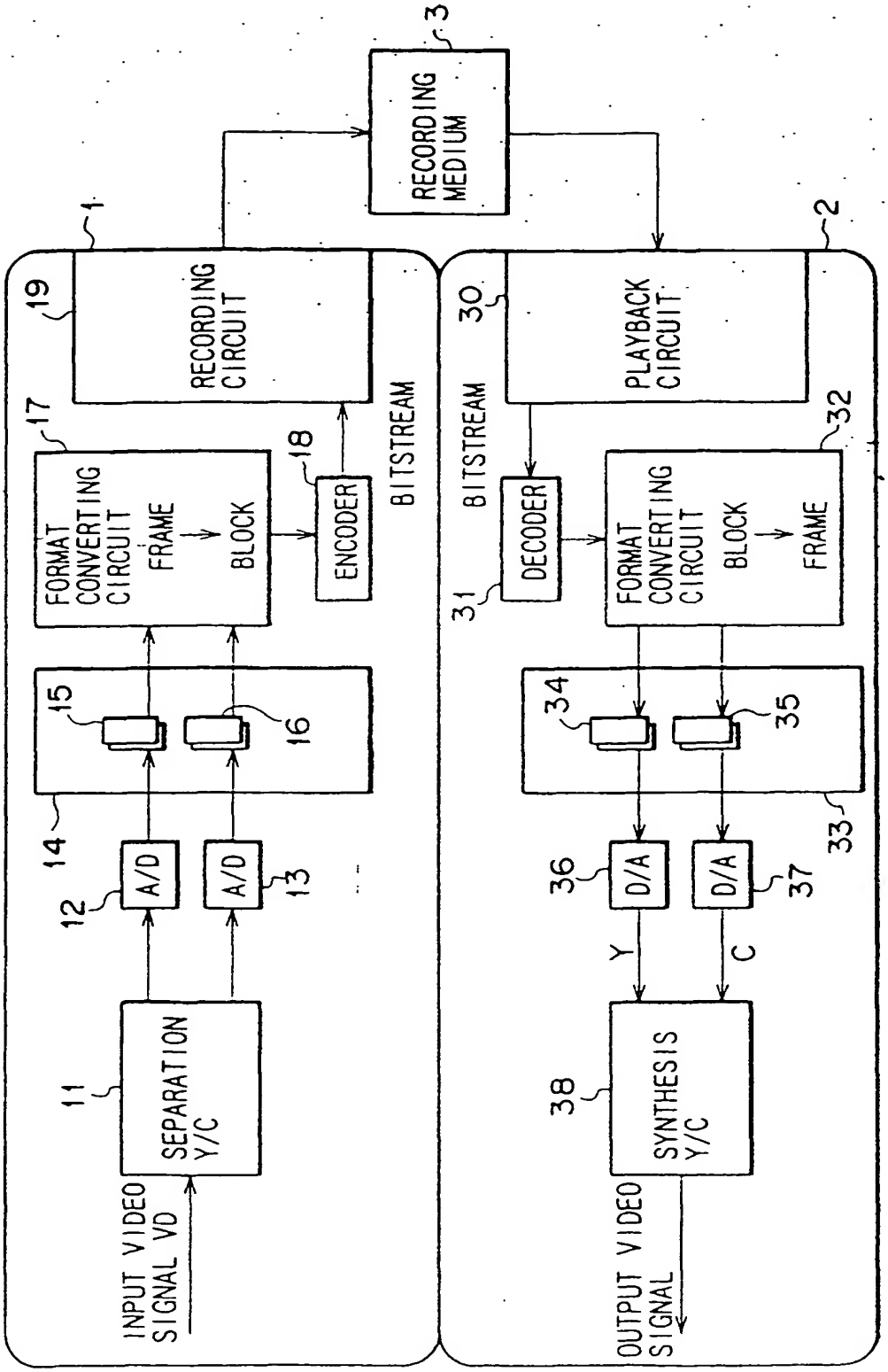

FRAME NEVER SUBJECTED TO FORWARD & BACKWARD PREDICTION ENCODING (INTRA-PICTURE ENCODING)
FRAME NEVER SUBJECTED TO FORWARD & BACKWARD PREDICTION ENCODING (INTER-PICTURE ENCODING)
FRAME NEVER SUBJECTED TO FORWARD & BACKWARD PREDICTION ENCODING (INTER-PICTURE ENCODING)
FRAME POSSIBLY SUBJECTED TO FORWARD & BACKWARD PREDICTION ENCODING
FRAME POSSIBLY SUBJECTED TO FORWARD & BACKWARD PREDICTION ENCODING
MOVING-PICTURE VIDEO SIGNAL VD
TRANSMITTED DATA DATA
TRANSMITTED DATA OF A FRAME NEVER SUBJECTED TO FORWARD & BACKWARD PREDICTION ENCODING
TRANSMITTED DATA OF A FRAME POSSIBLY SUBJECTED TO FORWARD & BACKWARD PREDICTION ENCODING
F1
F2
F3
F4
F5
F1X
F2X
F3X
F4X
F5X
X1
X2
X3
X4
X5
X6
SP1
SP2
SP3
SP4
(I)
(B)
(P)
(B)
(P)

FIG. 5


RECORDING MEDIUM
3
1
2
19
RECORDING CIRCUIT
30
PLAYBACK CIRCUIT
17
FORMAT CONVERTING CIRCUIT
FRAME → BLOCK
18
ENCODER
BITSTREAM
BITSTREAM
DECODER
31
32
FORMAT CONVERTING CIRCUIT
BLOCK → FRAME
15
16
14
34
35
33
12
A/D
13
A/D
36
D/A
37
D/A
11
SEPARATION Y/C
Y
C
38
SYNTHESIS Y/C
INPUT VIDEO SIGNAL VD
OUTPUT VIDEO SIGNAL

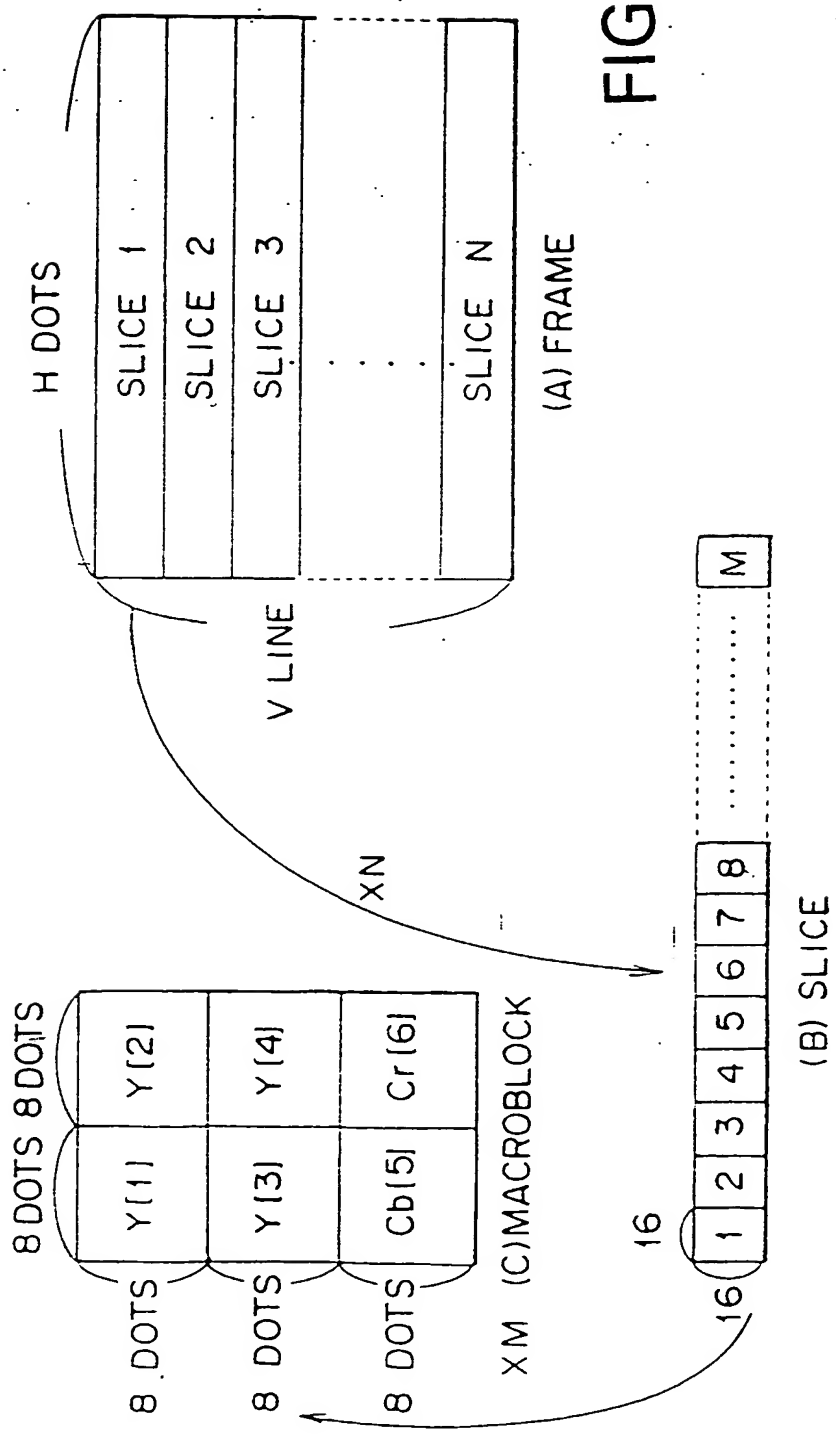


FIG. 6a

FIG. 6b

FIG. 6c

FIG. 7

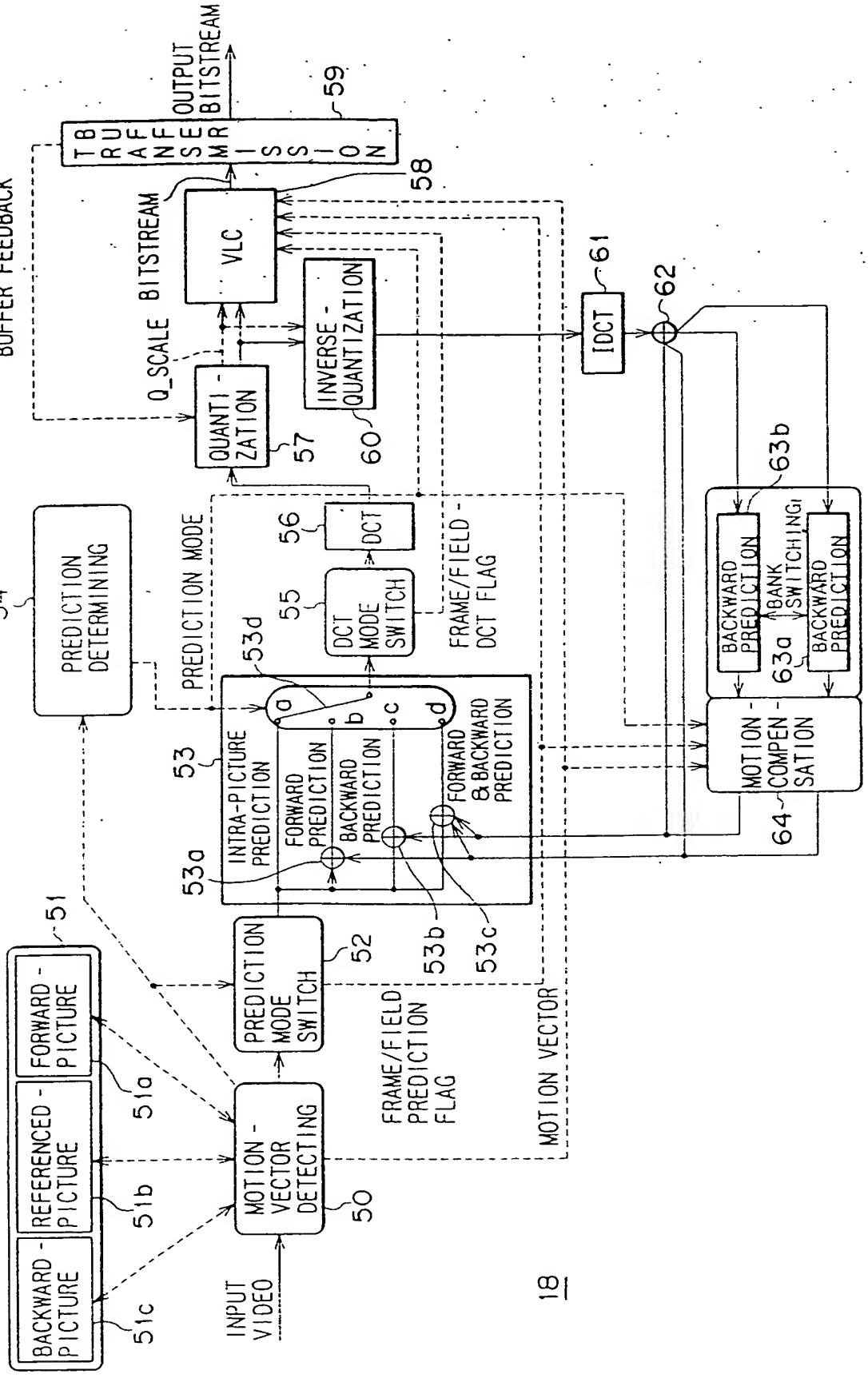

BUFFER FEEDBACK
OUTPUT BITSTREAM
TRANSMISSION
59
58
VLC
BITSTREAM
Q_SCALE
QUANTIZATION
57
INVERSE-QUANTIZATION
60
IDCT
61
62
PREDICTION DETERMINING
54
PREDICTION MODE
56
DCT
55
DCT MODE SWITCH
FRAME/FIELD-DCT FLAG
53d
53
INTRA-PICTURE PREDICTION
FORWARD PREDICTION
BACKWARD PREDICTION
FORWARD & BACKWARD PREDICTION
53a
53b
53c
BACKWARD PREDICTION
BANK SWITCHING
63a
63b
MOTION-COMPENSATION
64
PREDICTION MODE SWITCH
52
FRAME/FIELD PREDICTION FLAG
MOTION VECTOR
MOTION-VECTOR DETECTING
50
INPUT VIDEO
BACKWARD-PICTURE
REFERENCED-PICTURE
FORWARD-PICTURE
51
51a
51b
51c
18

FIG.8

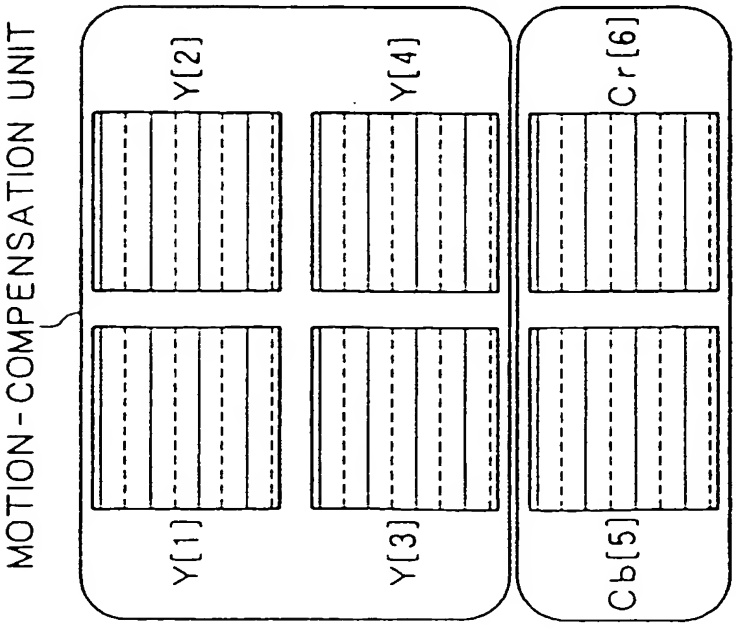

MOTION-COMPENSATION UNIT
Y[1]
Y[2]
Y[3]
Y[4]
Cb[5]
Cr[6]
FRAME-PREDICTION MODE
DATA OF A FIRST FIELD
DATA OF A SECOND FIELD

FIG.9

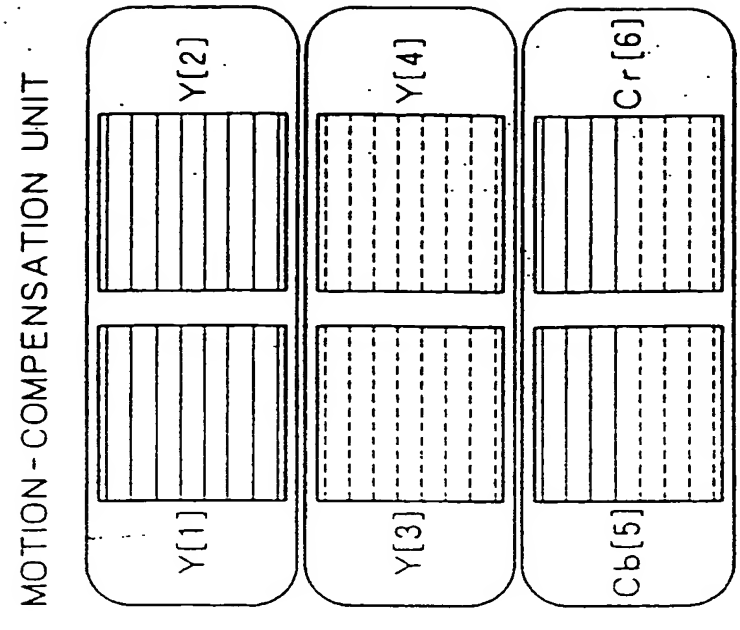

MOTION-COMPENSATION UNIT
Y[1]
Y[2]
Y[3]
Y[4]
Cb[5]
Cr[6]
FIELD-PREDICTION MODE
DATA OF A FIRST FIELD
DATA OF A SECOND FIELD

FIG.10

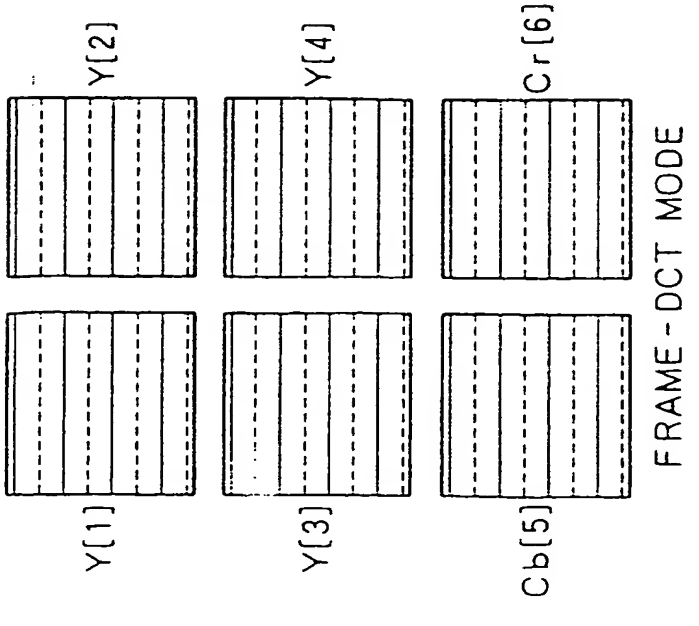

Y[1]
Y[2]
Y[3]
Y[4]
Cb[5]
Cr[6]
FRAME-DCT MODE
DATA OF A FIRST FIELD
DATA OF A SECOND FIELD

FIG.11

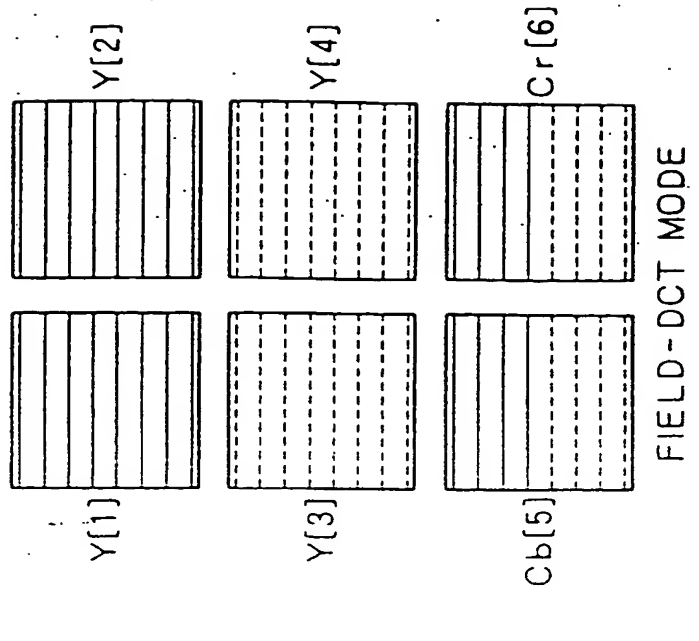

Y[1]
Y[2]
Y[3]
Y[4]
Cb[5]
Cr[6]
FIELD-DCT MODE
DATA OF A FIRST FIELD
DATA OF A SECOND FIELD

FIG. 12

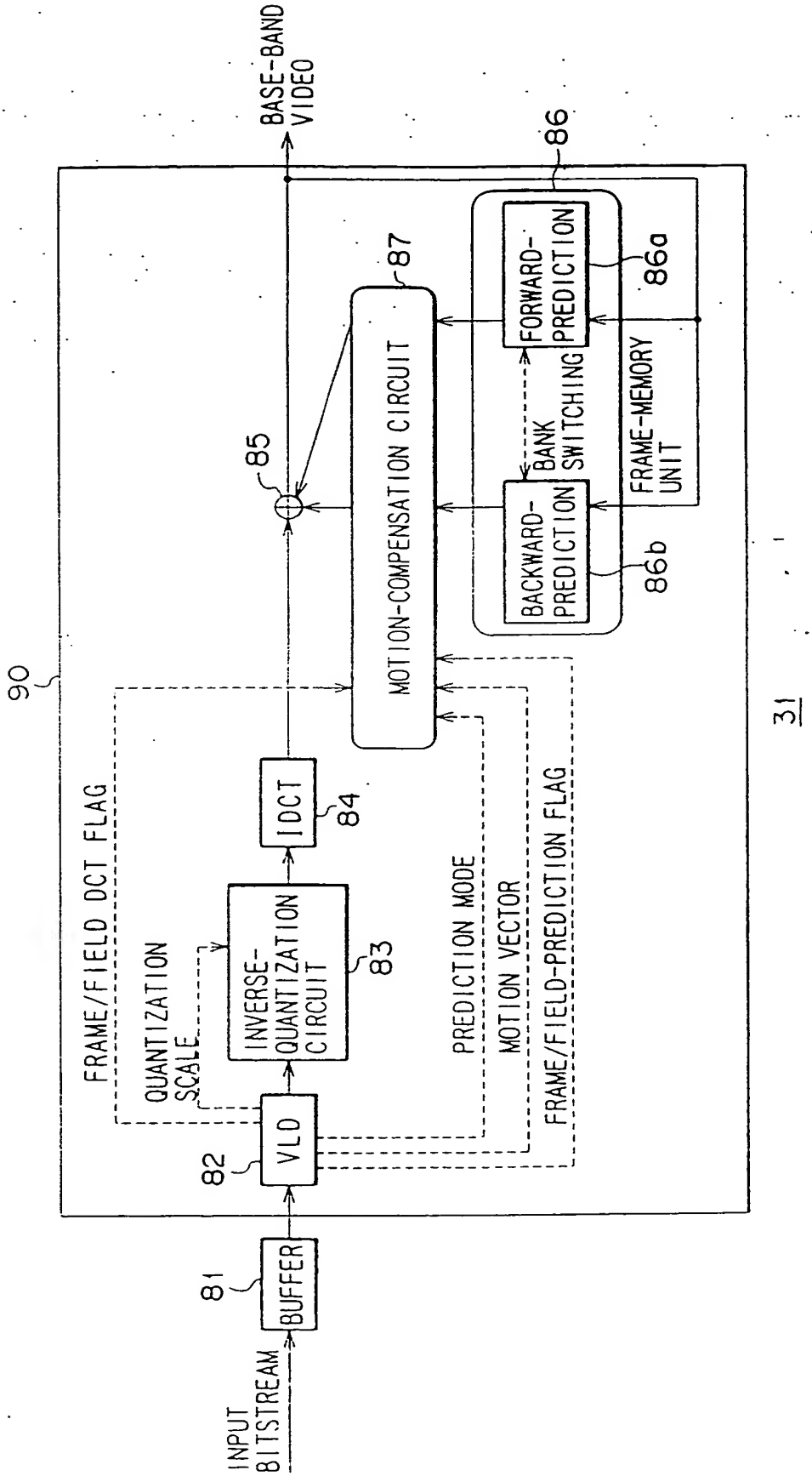

INPUT BITSTREAM
81
BUFFER
82
VLD
QUANTIZATION SCALE
INVERSE-QUANTIZATION CIRCUIT
83
IDCT
84
FRAME/FIELD DCT FLAG
90
85
MOTION-COMPENSATION CIRCUIT
87
PREDICTION MODE
MOTION VECTOR
FRAME/FIELD-PREDICTION FLAG
BASE-BAND VIDEO
86
FORWARD-PREDICTION
86a
BANK SWITCHING
FRAME-MEMORY UNIT
BACKWARD-PREDICTION
86b
31

FIG. 13

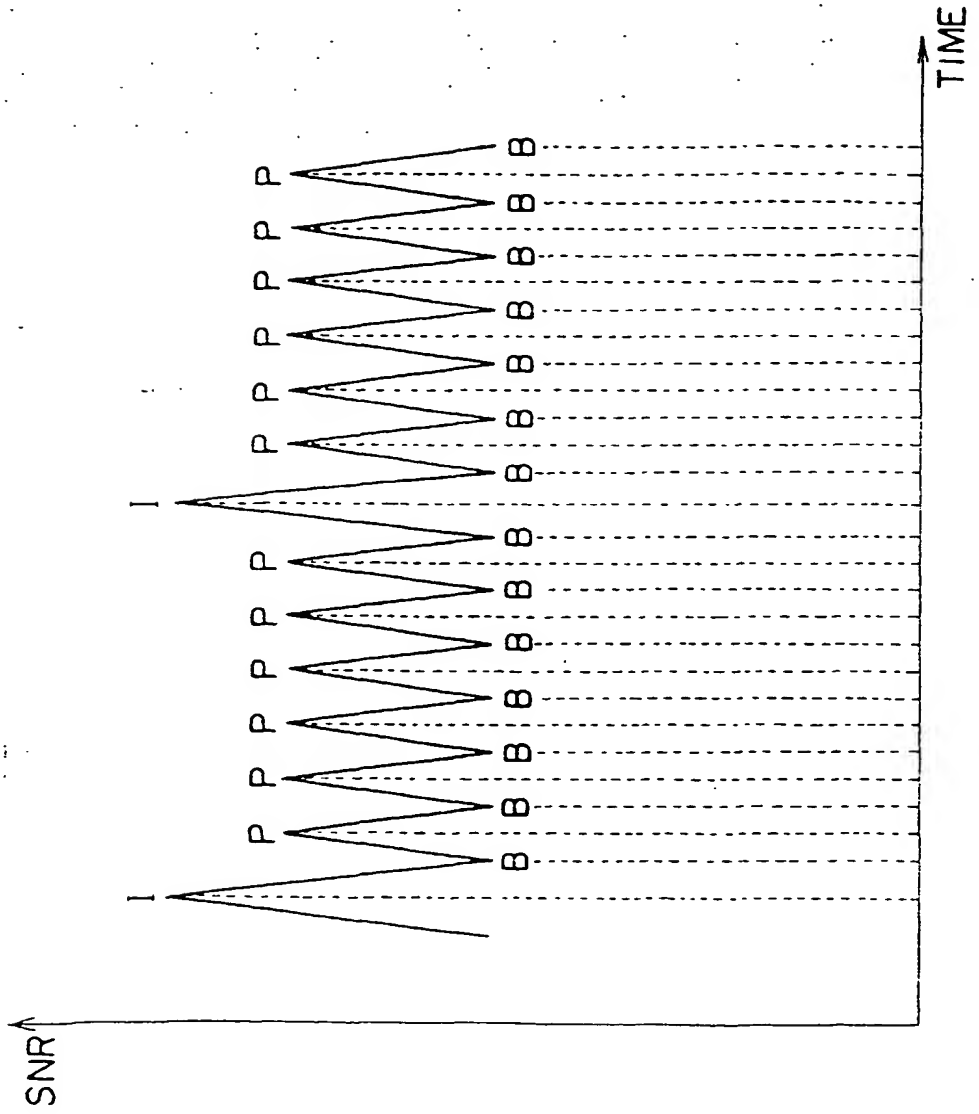


FIG. 14

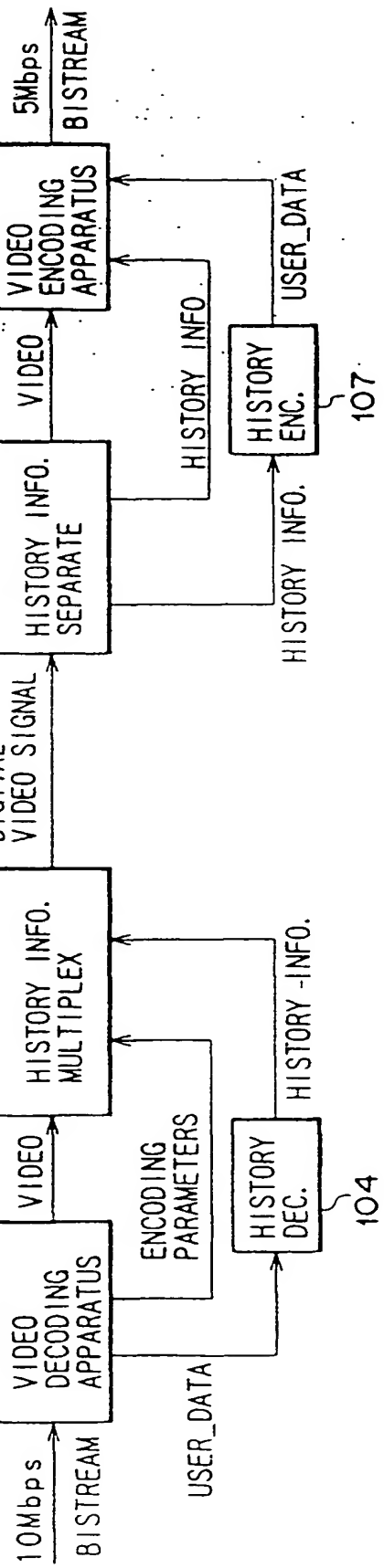

10Mbps
BISTREAM
102
VIDEO DECODING APPARATUS
VIDEO
ENCODING PARAMETERS
USER_DATA
HISTORY DEC.
104
HISTORY -INFO.
103
HISTORY INFO. MULTIPLEX
BASE - BAND DIGITAL VIDEO SIGNAL
105
HISTORY INFO. SEPARATE
VIDEO
HISTORY INFO.
HISTORY ENC.
107
HISTORY INFO.
USER_DATA
106
VIDEO ENCODING APPARATUS
5Mbps
BISTREAM
101

FIG.15

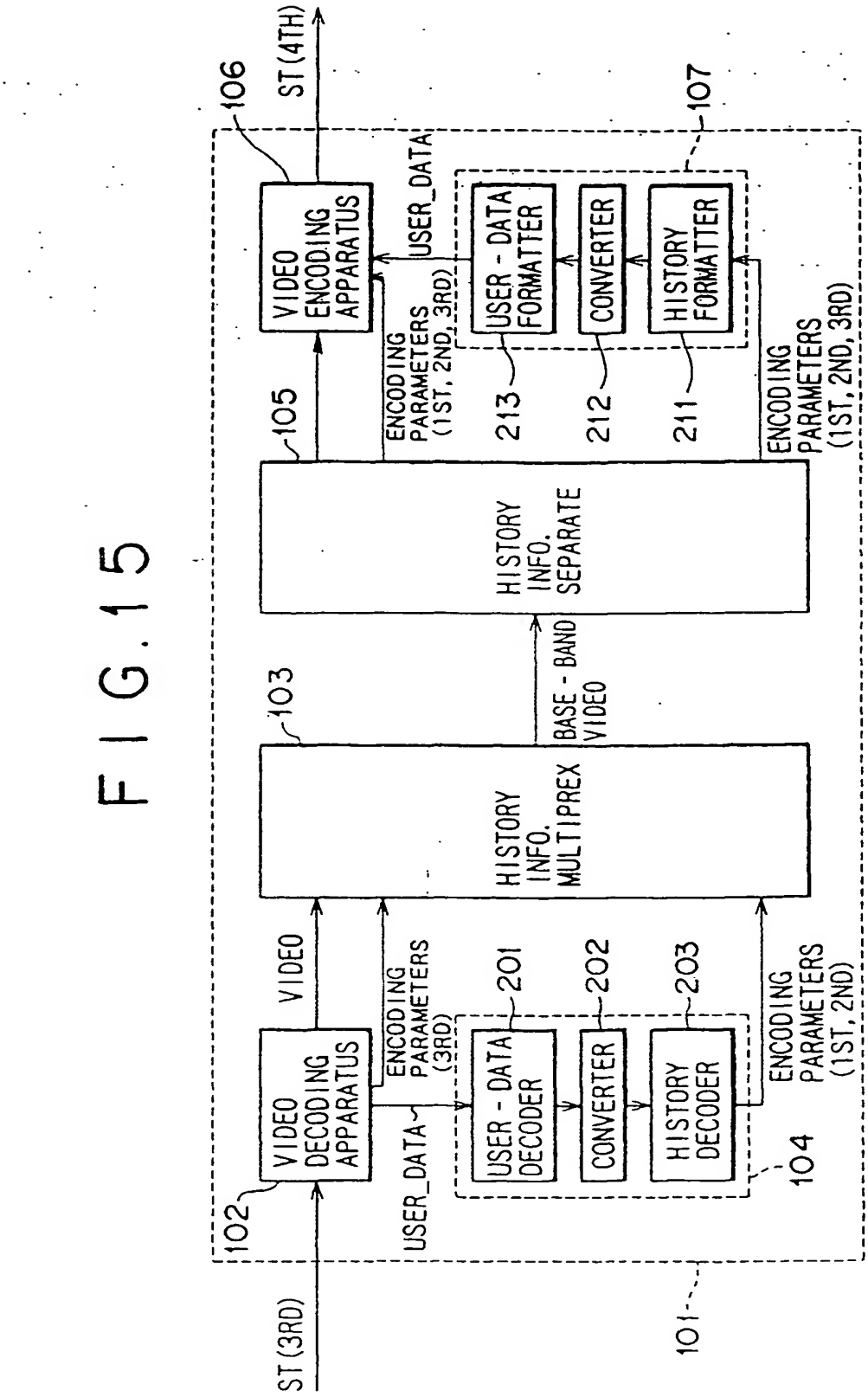

ST(3RD)
101
102
VIDEO DECODING APPARATUS
VIDEO
USER_DATA
ENCODING PARAMETERS (3RD)
201
USER - DATA DECODER
202
CONVERTER
203
HISTORY DECODER
104
ENCODING PARAMETERS (1ST, 2ND)
103
HISTORY INFO. MULTIPREX
BASE - BAND VIDEO
105
HISTORY INFO. SEPARATE
ENCODING PARAMETERS (1ST, 2ND, 3RD)
211
212
213
107
USER - DATA FORMATTER
CONVERTER
HISTORY FORMATTER
USER_DATA
106
VIDEO ENCODING APPARATUS
ST(4TH)

FIG. 16

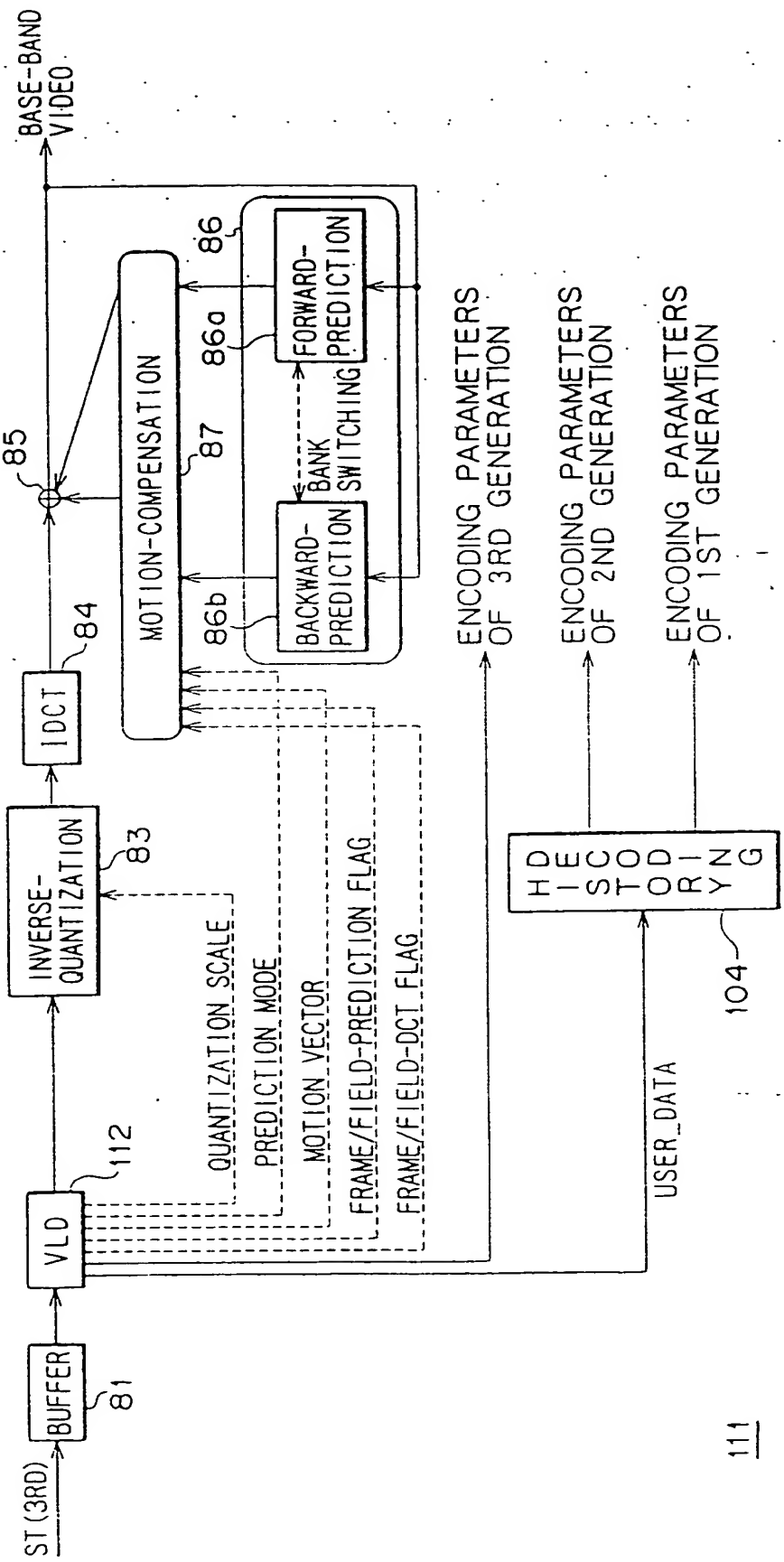

ST (3RD)
BUFFER
81
VLD
112
INVERSE-QUANTIZATION
83
IDCT
84
85
MOTION-COMPENSATION
87
86
86a
86b
FORWARD-PREDICTION
BACKWARD-PREDICTION
BANK SWITCHING
BASE-BAND VIDEO
QUANTIZATION SCALE
PREDICTION MODE
MOTION VECTOR
FRAME/FIELD-PREDICTION FLAG
FRAME/FIELD-DCT FLAG
ENCODING PARAMETERS OF 3RD GENERATION
ENCODING PARAMETERS OF 2ND GENERATION
ENCODING PARAMETERS OF 1ST GENERATION
HDI-ECTOODRI-YNG
104
USER_DATA
111

FIG. 19

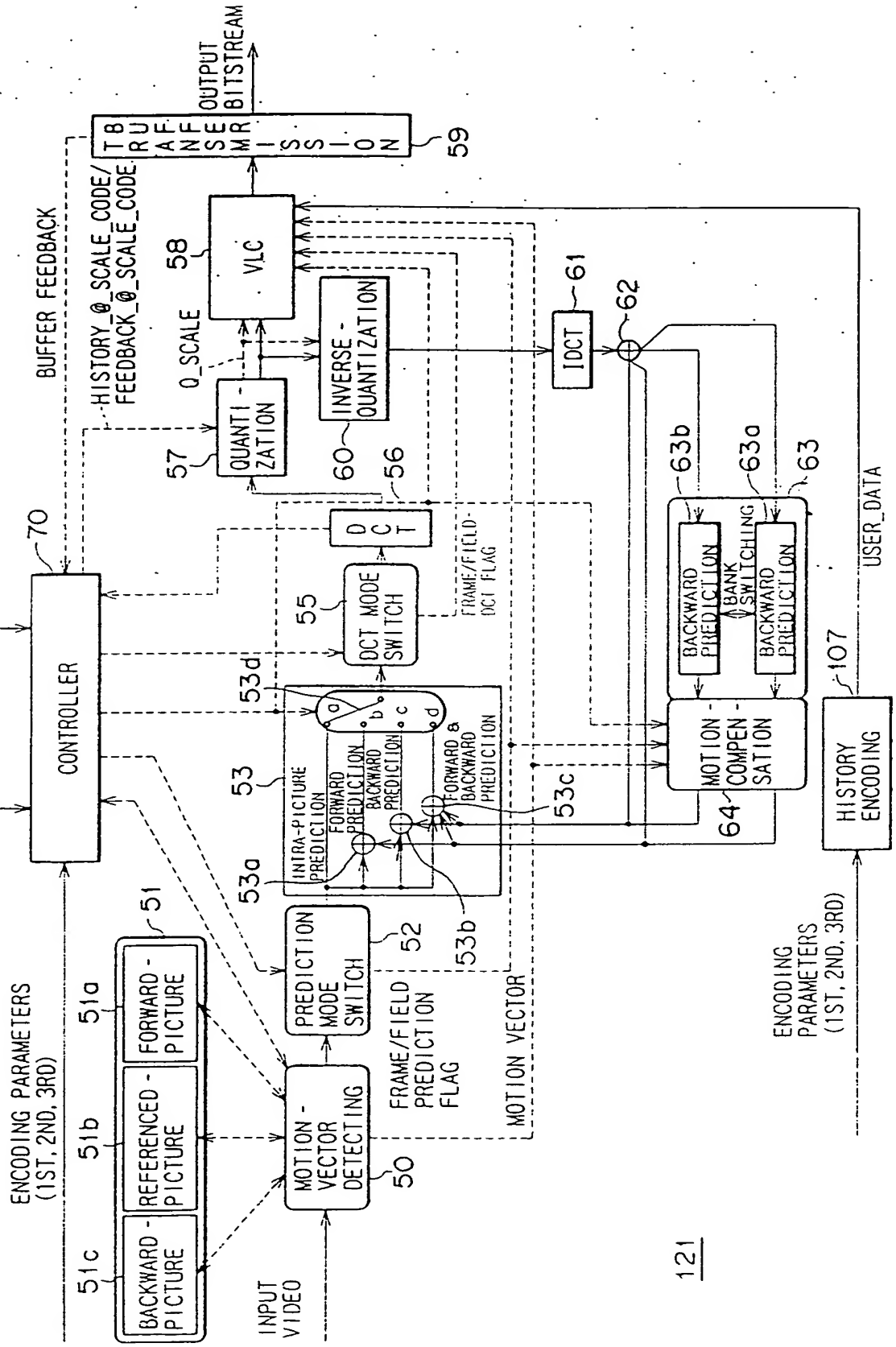

OUTPUT BITSTREAM
TRANSMISSION BUFFER
59
VLC
58
HISTORY_@_SCALE_CODE/
FEEDBACK_@_SCALE_CODE
BUFFER FEEDBACK
Q_SCALE
QUANTI-ZATION
57
INVERSE-QUANTIZATION
60
56
IDCT
61
62
DCT
DCT MODE SWITCH
55
FRAME/FIELD-DCT FLAG
CONTROLLER
70
GOP_STRUCTURE
TARGET_BIT_RATE
53d
53
INTRA-PICTURE PREDICTION
FORWARD PREDICTION
BACKWARD PREDICTION
FORWARD & BACKWARD PREDICTION
53a
53b
53c
63b
63a
63
BACKWARD PREDICTION
BANK SWITCHING
BACKWARD PREDICTION
MOTION-COMPEN-SATION
64
HISTORY ENCODING
107
USER_DATA
ENCODING PARAMETERS (1ST, 2ND, 3RD)
51
51a
FORWARD-PICTURE
51b
REFERENCED-PICTURE
51c
BACKWARD-PICTURE
PREDICTION MODE SWITCH
52
FRAME/FIELD PREDICTION FLAG
MOTION VECTOR
MOTION-VECTOR DETECTING
50
INPUT VIDEO
ENCODING PARAMETERS (1ST, 2ND, 3RD)
121

FIG. 20

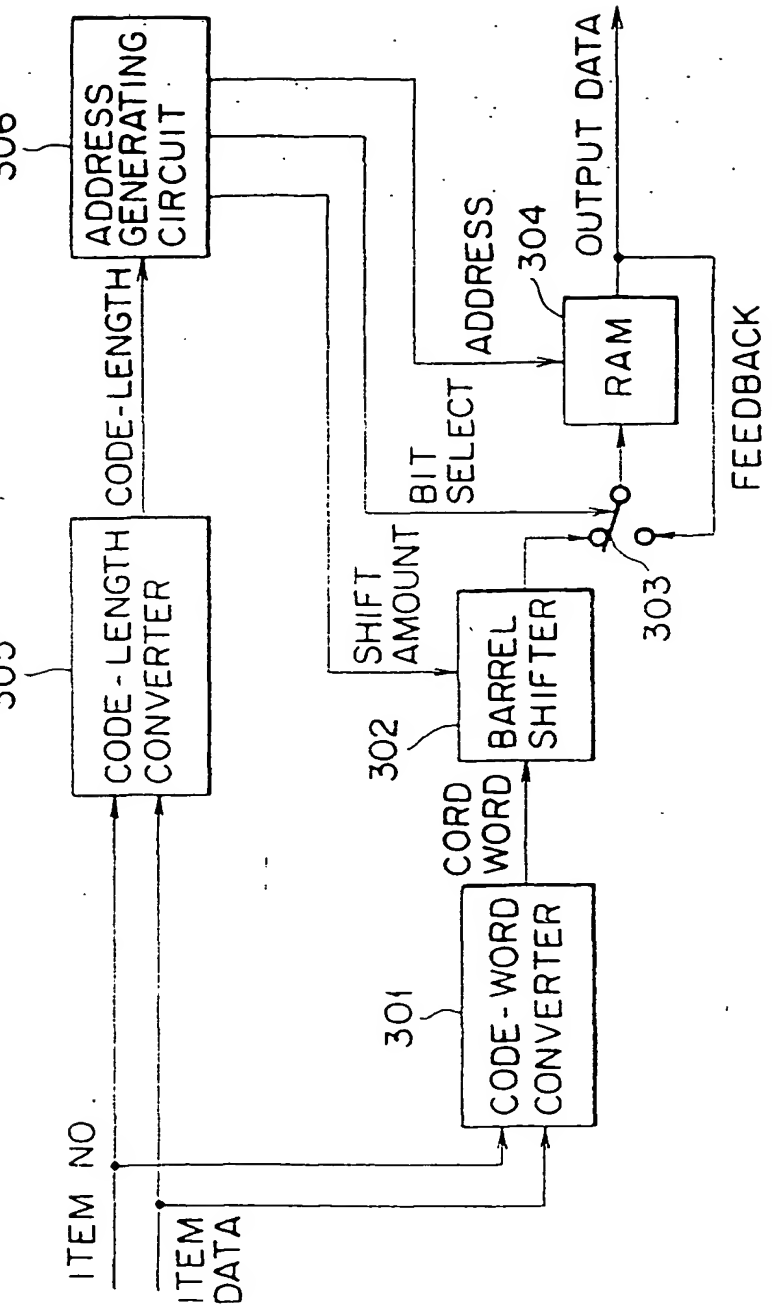

306
ADDRESS GENERATING CIRCUIT
CODE-LENGTH
305
CODE-LENGTH CONVERTER
ITEM NO
ITEM DATA
SHIFT AMOUNT
BIT SELECT
ADDRESS
304
OUTPUT DATA
RAM
FEEDBACK
303
302
BARREL SHIFTER
CORD WORD
301
CODE-WORD CONVERTER
211

FIG. 21

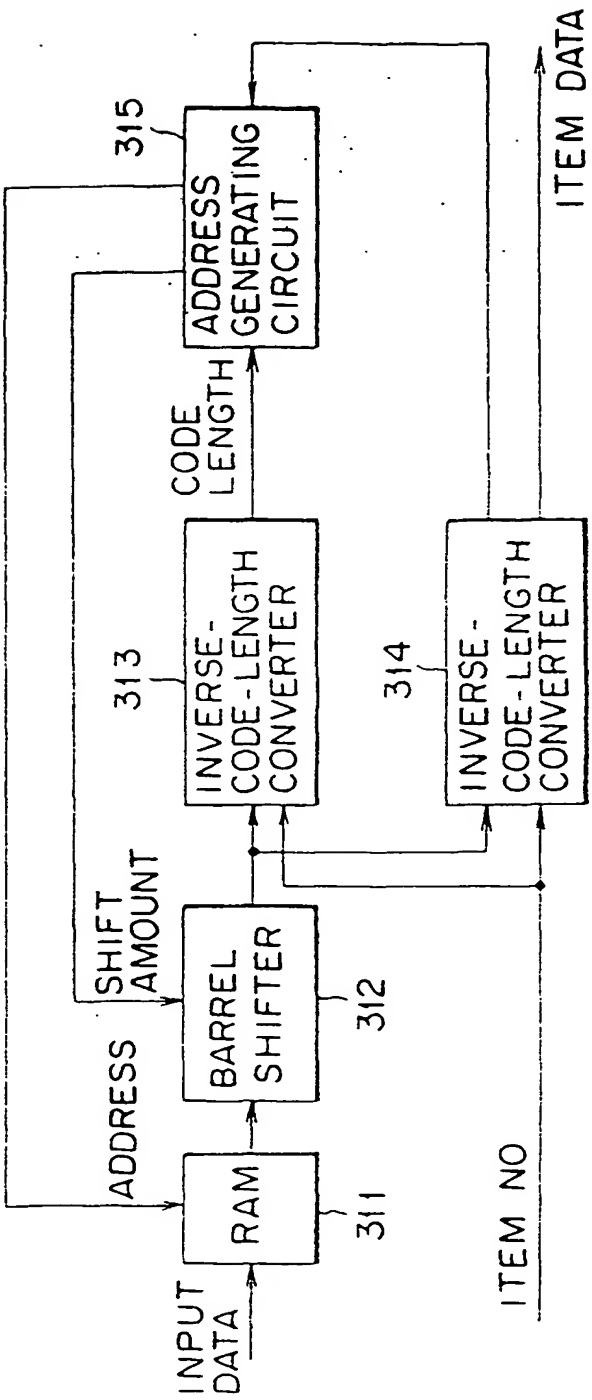

INPUT DATA
RAM
311
ADDRESS
BARREL SHIFTER
312
SHIFT AMOUNT
313
INVERSE-CODE-LENGTH CONVERTER
CODE LENGTH
315
ADDRESS GENERATING CIRCUIT
314
INVERSE-CODE-LENGTH CONVERTER
ITEM NO
ITEM DATA
203

FIG. 22

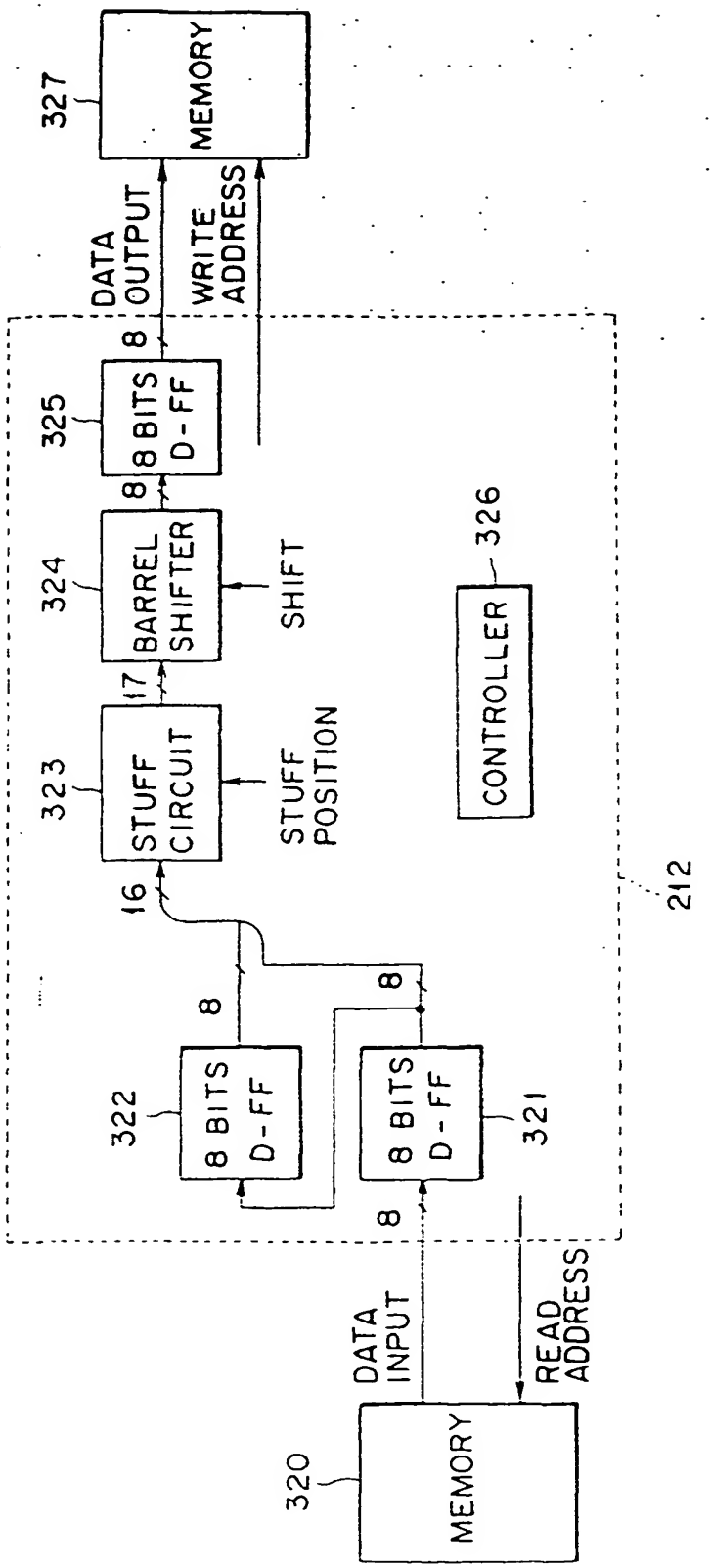

327
MEMORY
DATA OUTPUT
WRITE ADDRESS
8
325
8 BITS D-FF
8
324
BARREL SHIFTER
SHIFT
326
CONTROLLER
17
323
STUFF CIRCUIT
STUFF POSITION
16
212
8
8
322
8 BITS D-FF
8 BITS D-FF
321
8
DATA INPUT
READ ADDRESS
320
MEMORY

FIG. 23

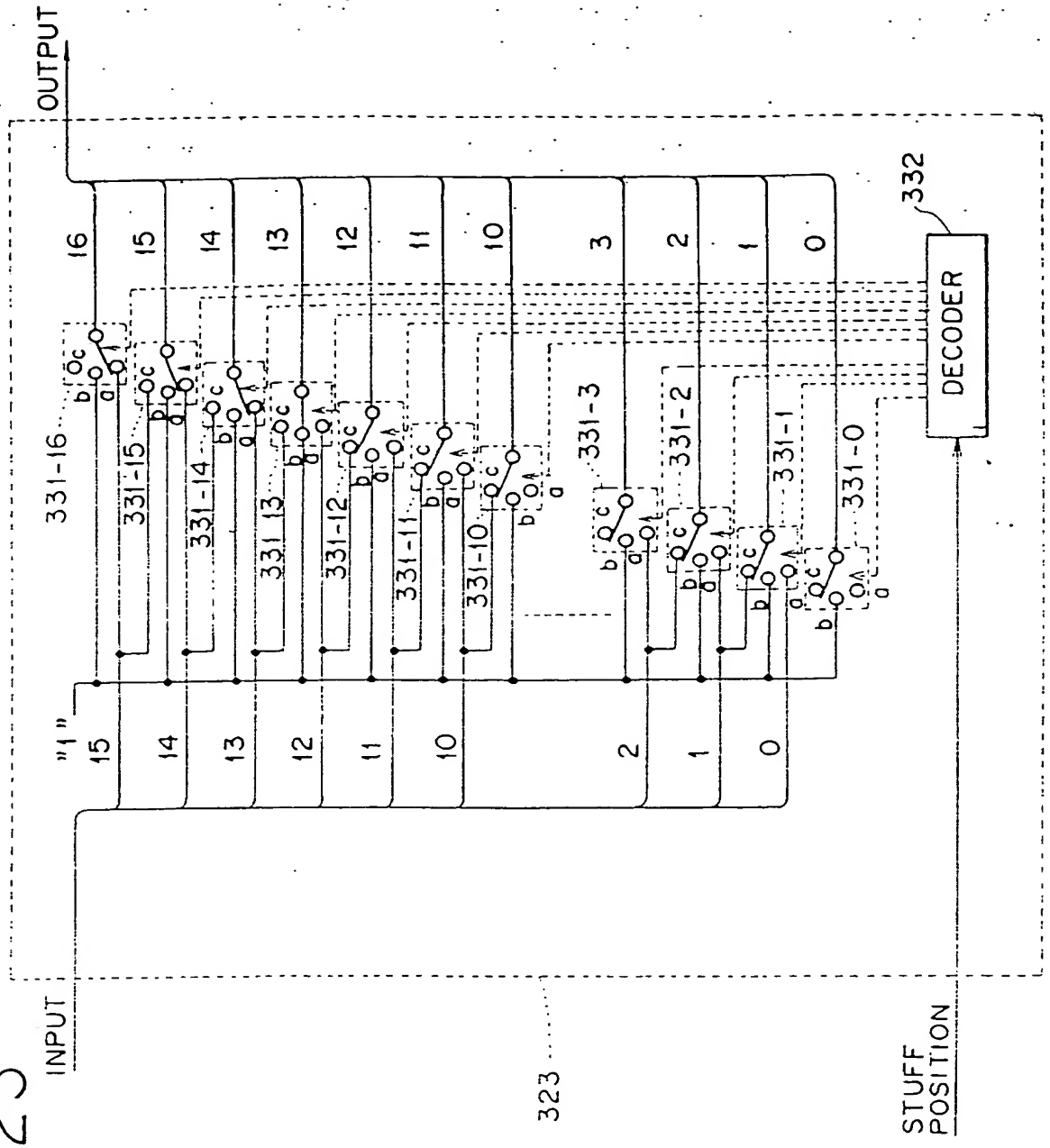

OUTPUT
INPUT
"1"
323
STUFF POSITION
DECODER
332
331-16
331-15
331-14
331 13
331-12
331-11
331-10
331-3
331-2
331-1
331-0
16
15
14
13
12
11
10
3
2
1
0
a
b
c

FIG.24

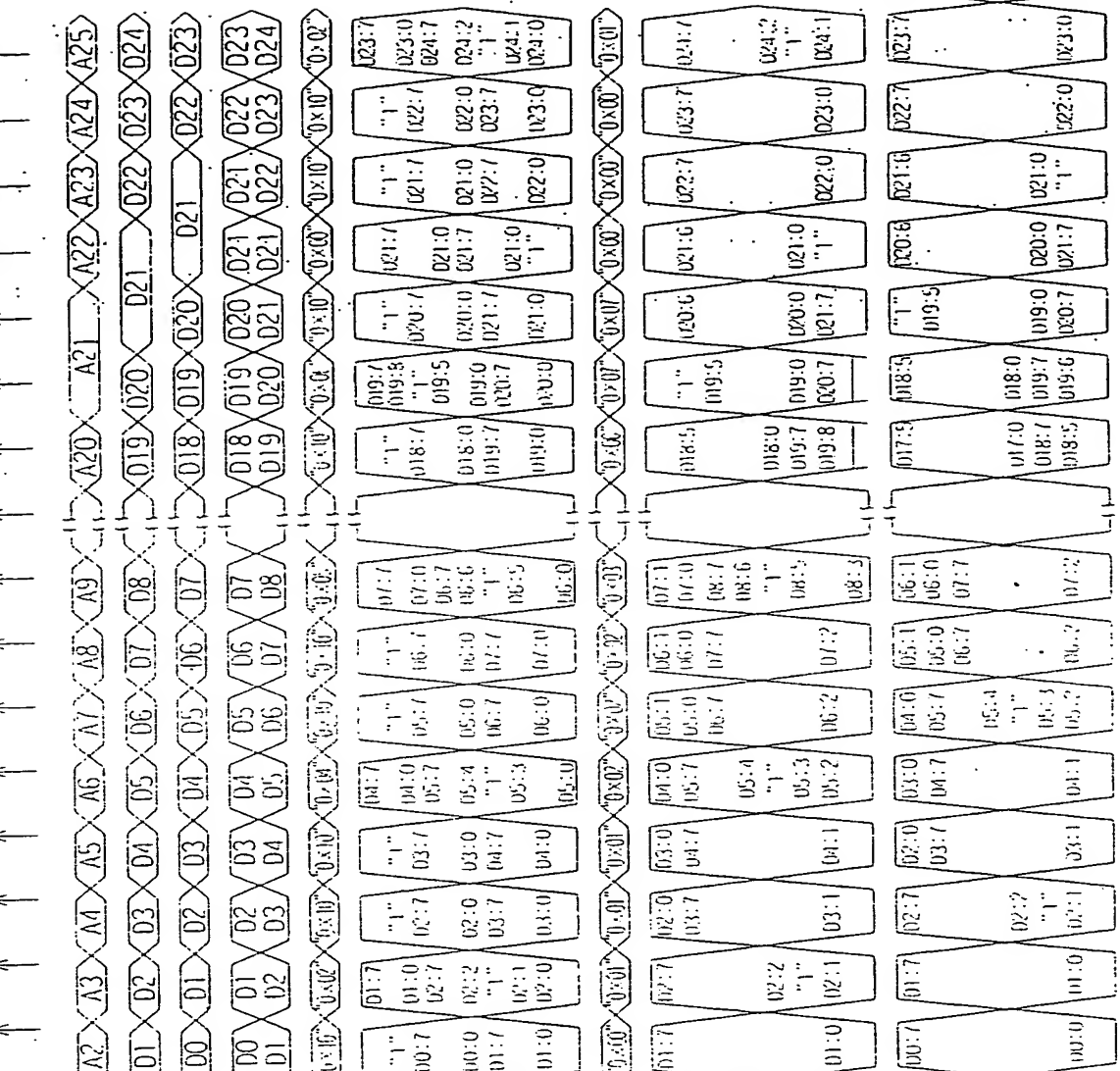

(A) Read Address
(B) D-FF 321 OUTPUT
(C) D-FF 322 OUTPUT
(D) STUFF CIRCUIT 323 INPUT
(E) stuff position
(F) STUFF CIRCUIT 323 OUTPUT
(G) shift
(H) BARREL SHIFTER 324 OUTPUT
(I) D-FF 325 OUTPUT

FIG. 25

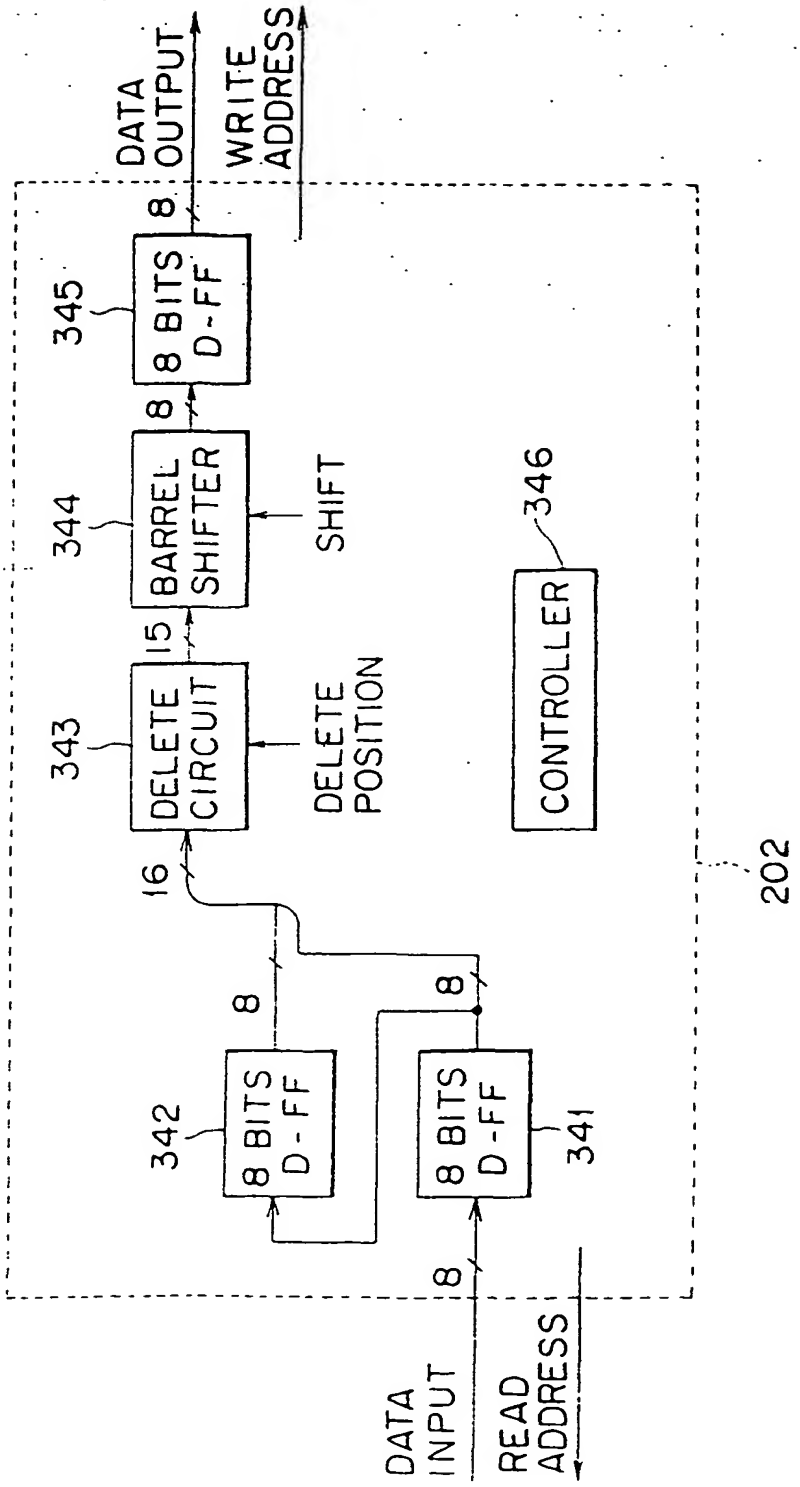

DATA OUTPUT
WRITE ADDRESS
345
8 BITS D-FF
8
344
BARREL SHIFTER
SHIFT
15
343
DELETE CIRCUIT
DELETE POSITION
16
346
CONTROLLER
202
8
342
8 BITS D-FF
8
8 BITS D-FF
341
8
DATA INPUT
READ ADDRESS

FIG.26

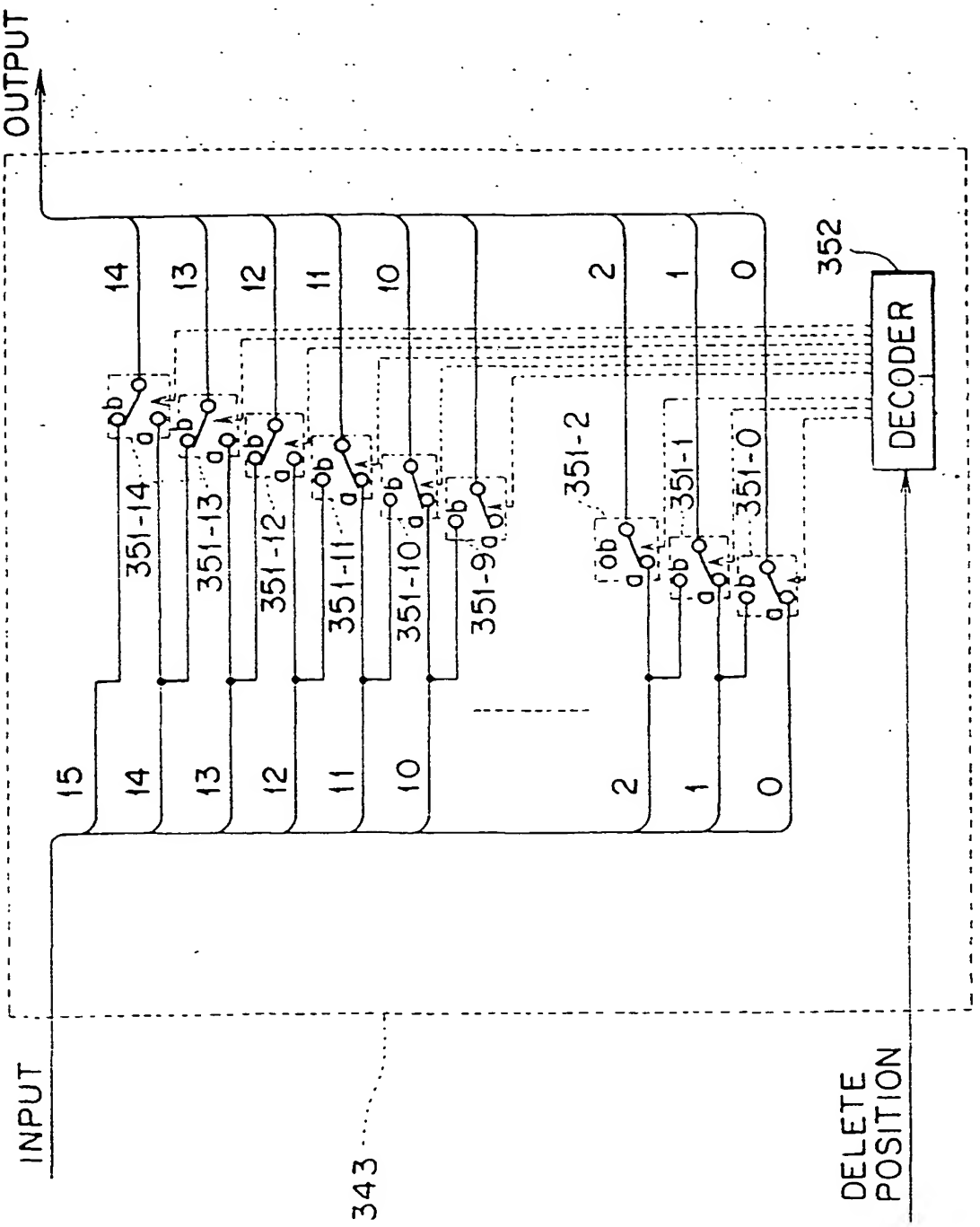

OUTPUT
INPUT
343
DELETE POSITION
DECODER
352
351-14
351-13
351-12
351-11
351-10
351-9
351-2
351-1
351-0
15
14
13
12
11
10
2
1
0
a
b
A

FIG. 27

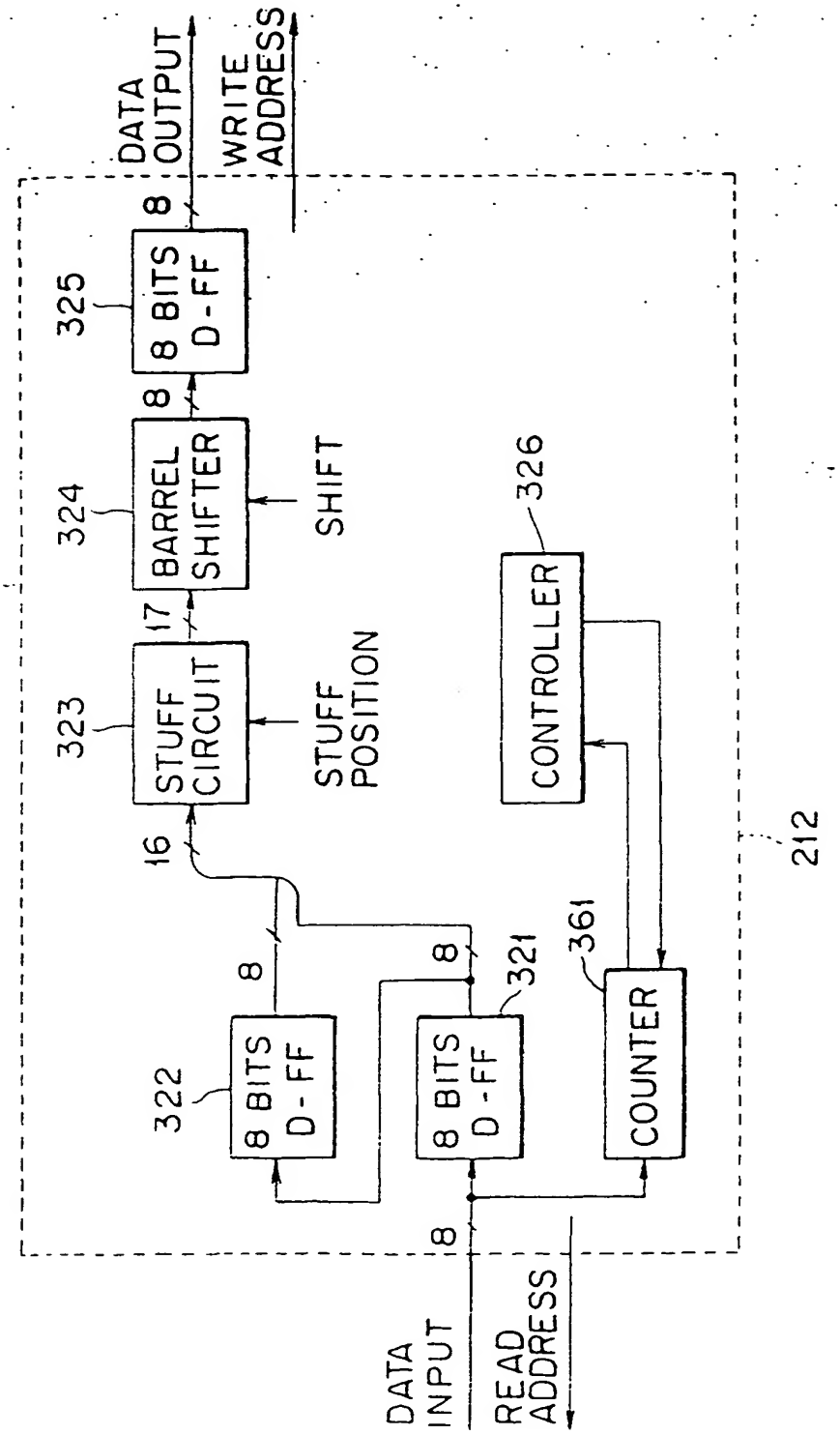

DATA OUTPUT
WRITE ADDRESS
8
325
8 BITS D-FF
8
324
BARREL SHIFTER
SHIFT
17
323
STUFF CIRCUIT
STUFF POSITION
16
326
CONTROLLER
212
8
322
8 BITS D-FF
8
321
8 BITS D-FF
361
COUNTER
8
DATA INPUT
READ ADDRESS

FIG. 28

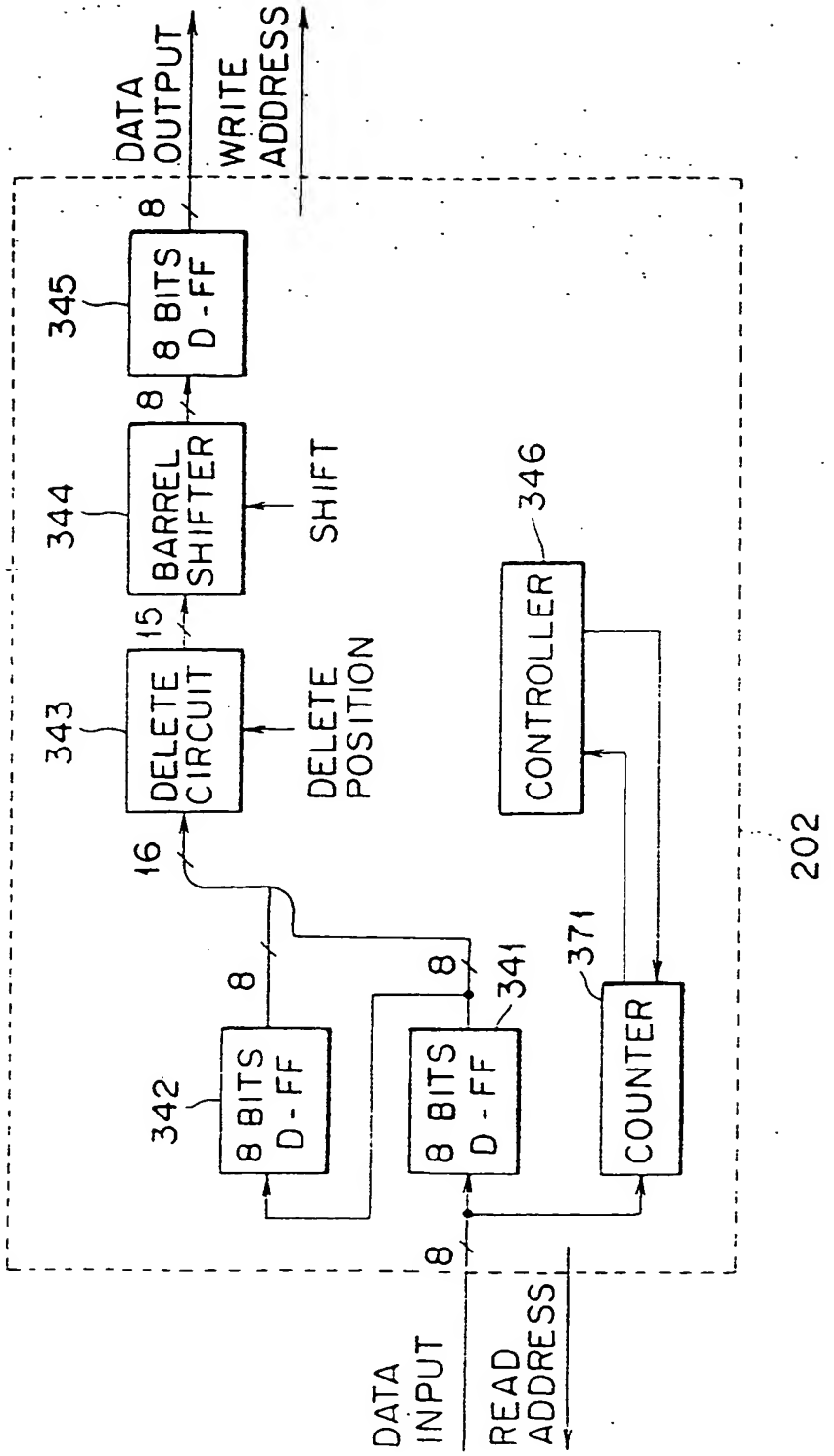

DATA OUTPUT
WRITE ADDRESS
8
345
8 BITS D-FF
8
344
BARREL SHIFTER
SHIFT
15
343
DELETE CIRCUIT
DELETE POSITION
16
346
CONTROLLER
342
8 BITS D-FF
8
8
341
8 BITS D-FF
371
COUNTER
8
DATA INPUT
READ ADDRESS
202

FIG. 29

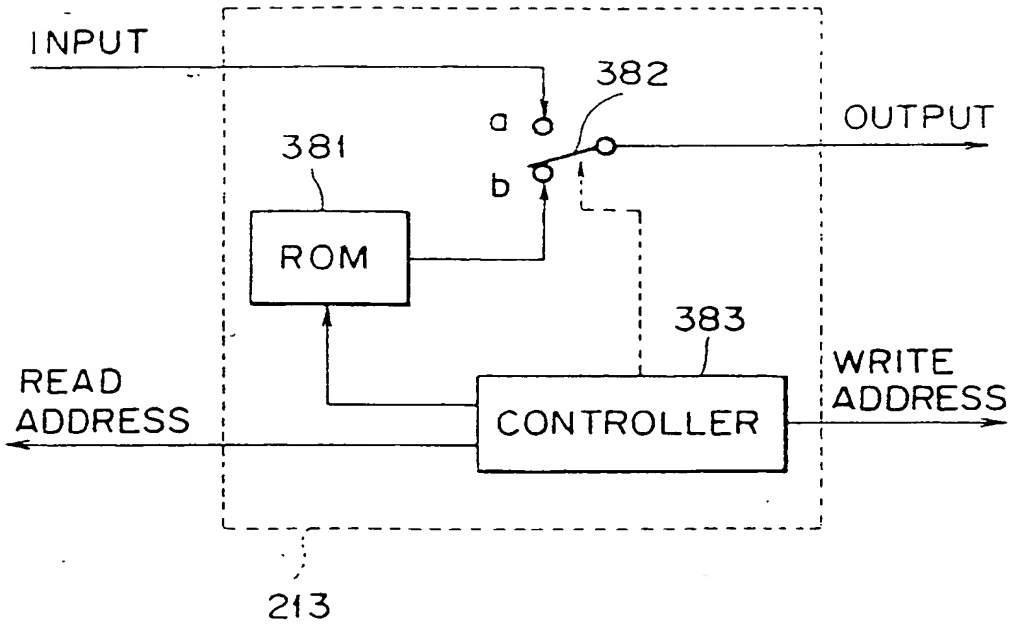

INPUT
382
a
OUTPUT
381
b
ROM
383
READ
ADDRESS
CONTROLLER
WRITE
ADDRESS
213

FIG. 30

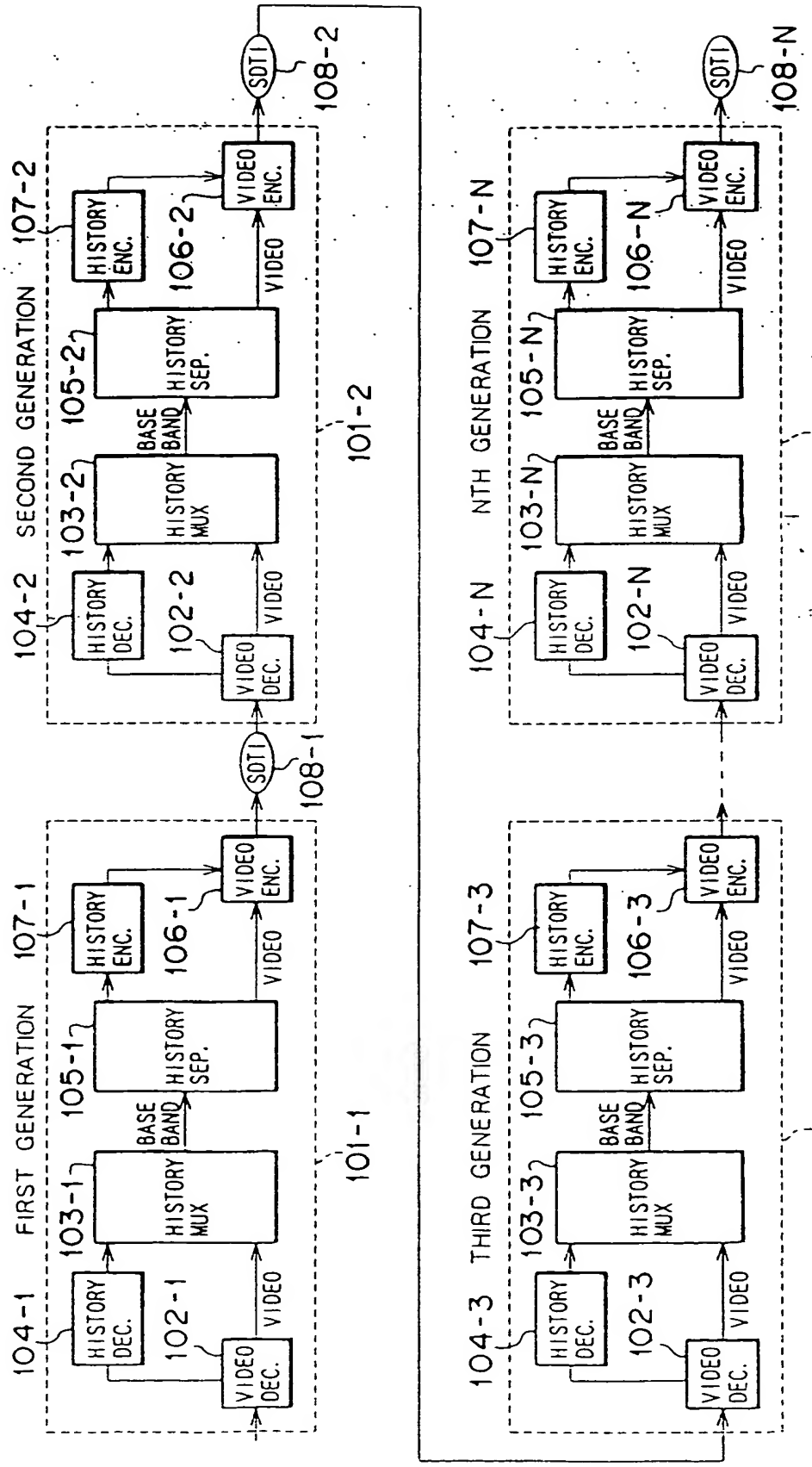

FIRST GENERATION
SECOND GENERATION
THIRD GENERATION
NTH GENERATION
HISTORY DEC.
VIDEO DEC.
HISTORY MUX
HISTORY SEP.
HISTORY ENC.
VIDEO ENC.
BASE BAND
VIDEO
SDTI
101-1
101-2
101-3
101-N
102-1
102-2
102-3
102-N
103-1
103-2
103-3
103-N
104-1
104-2
104-3
104-N
105-1
105-2
105-3
105-N
106-1
106-2
106-3
106-N
107-1
107-2
107-3
107-N
108-1
108-2
108-N

FIG.31

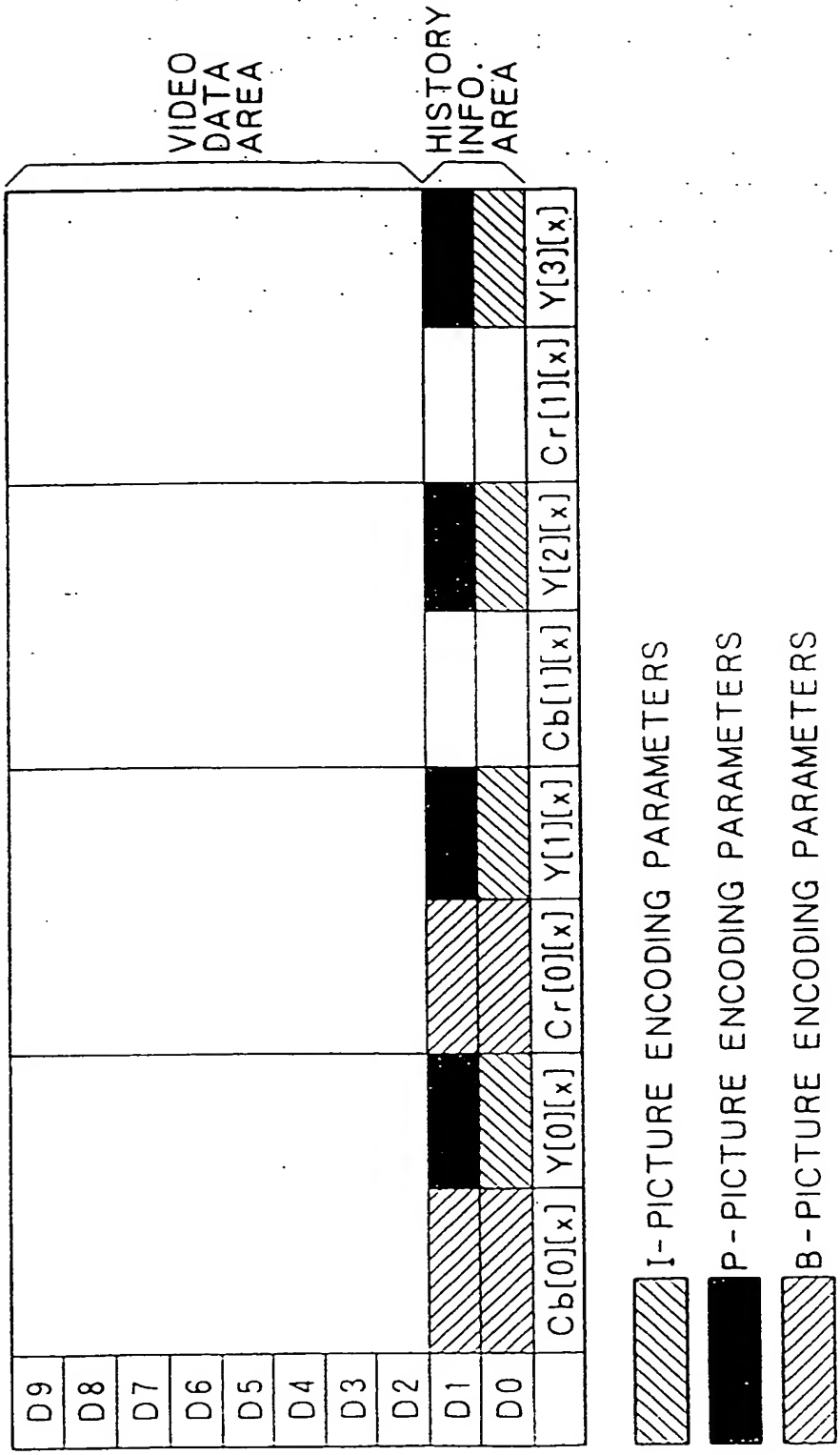

VIDEO DATA AREA
HISTORY INFO. AREA
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
Cb[0][x]
Y[0][x]
Cr[0][x]
Y[1][x]
Cb[1][x]
Y[2][x]
Cr[1][x]
Y[3][x]
I-PICTURE ENCODING PARAMETERS
P-PICTURE ENCODING PARAMETERS
B-PICTURE ENCODING PARAMETERS

FIG.32

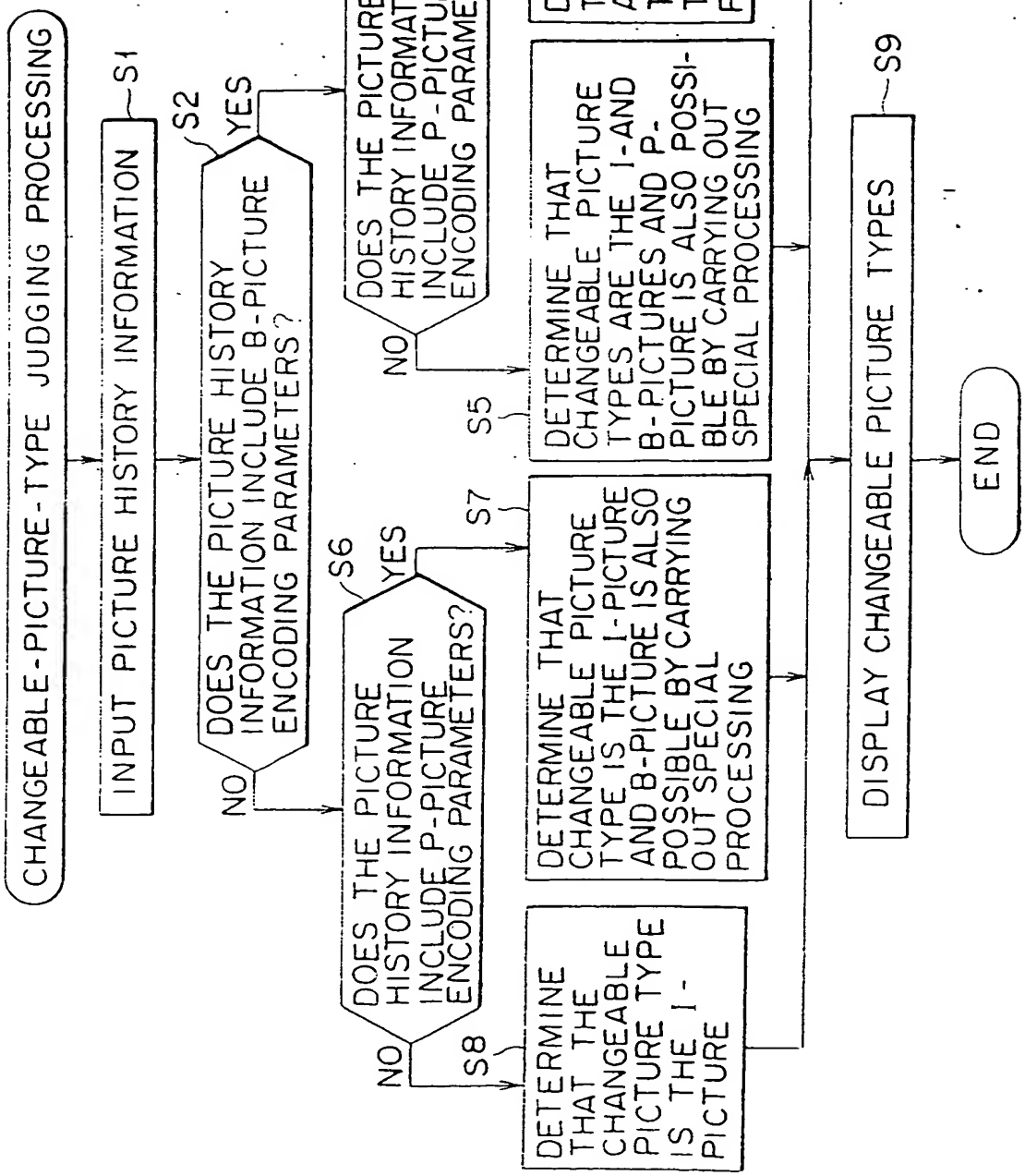

CHANGEABLE-PICTURE-TYPE JUDGING PROCESSING
INPUT PICTURE HISTORY INFORMATION
S1
DOES THE PICTURE HISTORY INFORMATION INCLUDE B-PICTURE ENCODING PARAMETERS?
S2
YES
NO
DOES THE PICTURE HISTORY INFORMATION INCLUDE P-PICTURE ENCODING PARAMETERS?
S3
YES
NO
DETERMINE THAT CHANGEABLE PICTURE TYPES ARE THE I-PICTURE
S4
DETERMINE THAT CHANGEABLE PICTURE TYPES ARE THE I-AND B-PICTURES AND P-PICTURE IS ALSO POSSIBLE BY CARRYING OUT SPECIAL PROCESSING
S5
DOES THE PICTURE HISTORY INFORMATION INCLUDE P-PICTURE ENCODING PARAMETERS?
S6
YES
NO
DETERMINE THAT CHANGEABLE PICTURE TYPE IS THE I-PICTURE AND B-PICTURE IS ALSO POSSIBLE BY CARRYING OUT SPECIAL PROCESSING
S7
DETERMINE THAT THE CHANGEABLE PICTURE TYPE IS THE I-PICTURE
S8
DISPLAY CHANGEABLE PICTURE TYPES
S9
END

FIG. 33

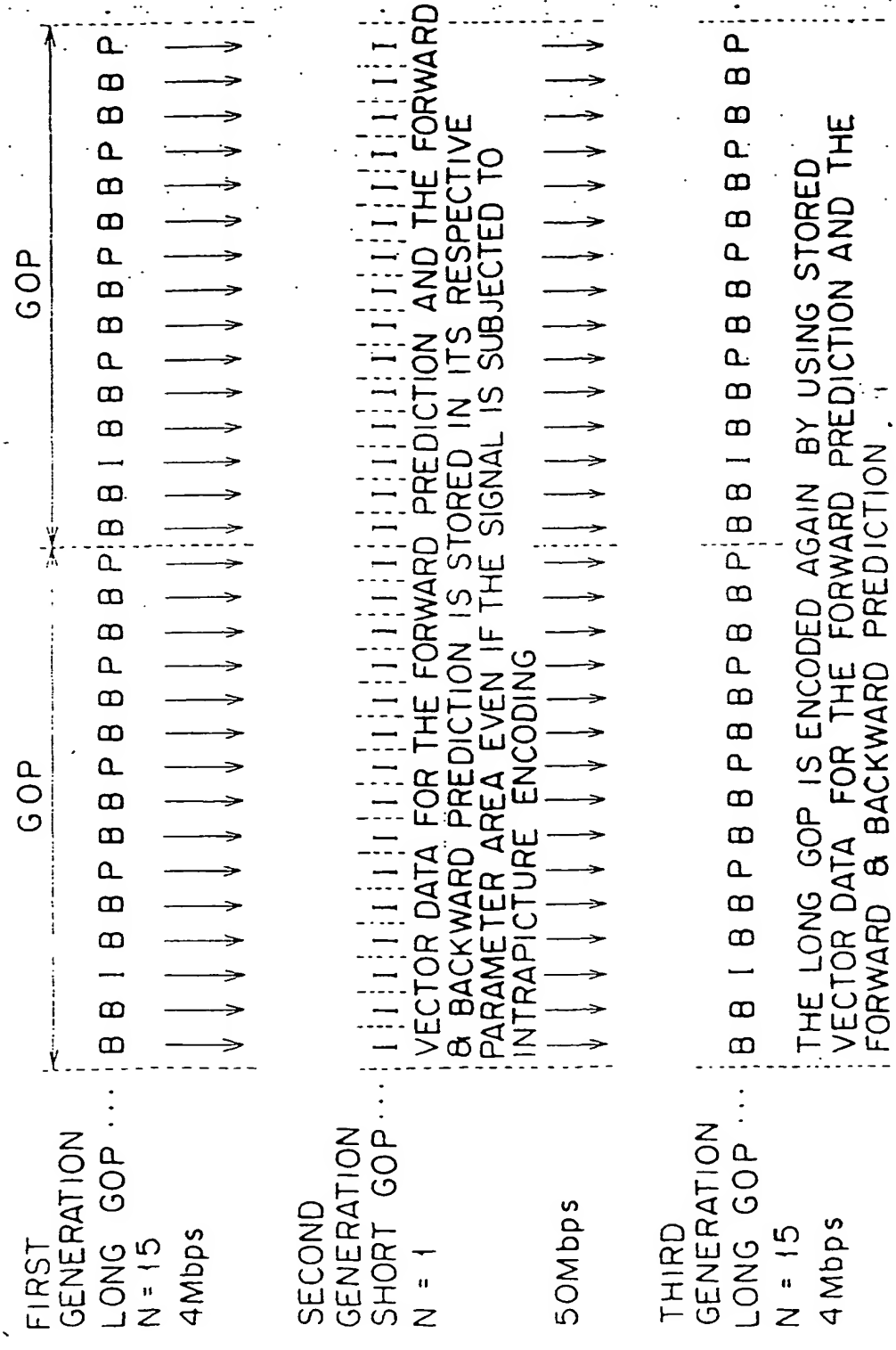

GOP
GOP
B B I B B P B B P B B P B B P B B P B B P ...
FIRST GENERATION
LONG GOP ...
N = 15
4Mbps
I I I I I I I I I I I I I I I I I I I I I ...
SECOND GENERATION
SHORT GOP ...
N = 1
50Mbps
VECTOR DATA FOR THE FORWARD PREDICTION AND THE FORWARD & BACKWARD PREDICTION IS STORED IN ITS RESPECTIVE PARAMETER AREA EVEN IF THE SIGNAL IS SUBJECTED TO INTRAPICTURE ENCODING
B B I B B P B B P B B P B B P B B P B B P ...
THIRD GENERATION
LONG GOP ...
N = 15
4Mbps
THE LONG GOP IS ENCODED AGAIN BY USING STORED VECTOR DATA FOR THE FORWARD PREDICTION AND THE FORWARD & BACKWARD PREDICTION

FIG. 34

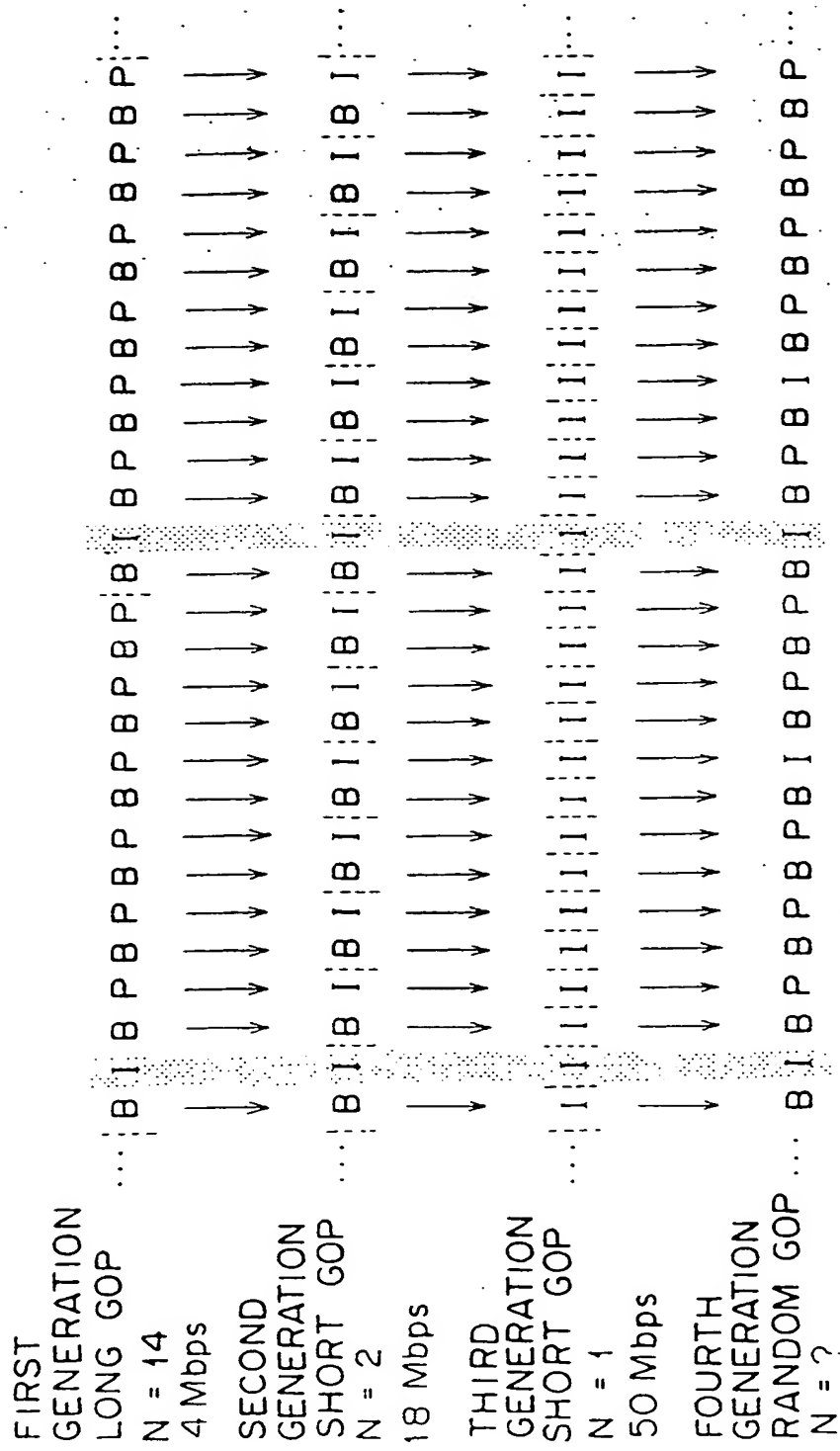
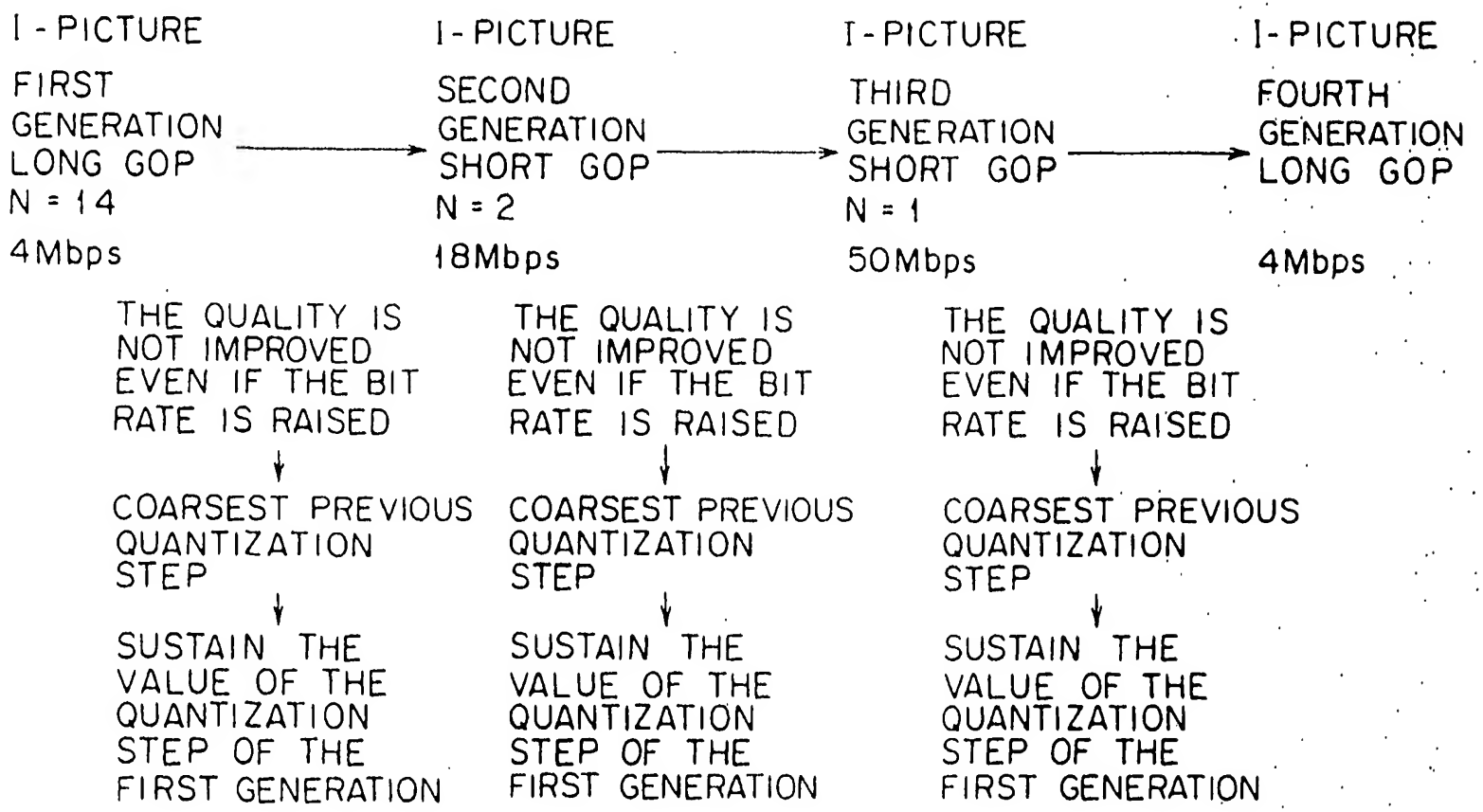

FIRST GENERATION LONG GOP N = 14 4 Mbps
SECOND GENERATION SHORT GOP N = 2 18 Mbps
THIRD GENERATION SHORT GOP N = 1 50 Mbps
FOURTH GENERATION RANDOM GOP N = ? 1 Mbps

FIG. 35



EP 0 942 605 A2

FIG. 36

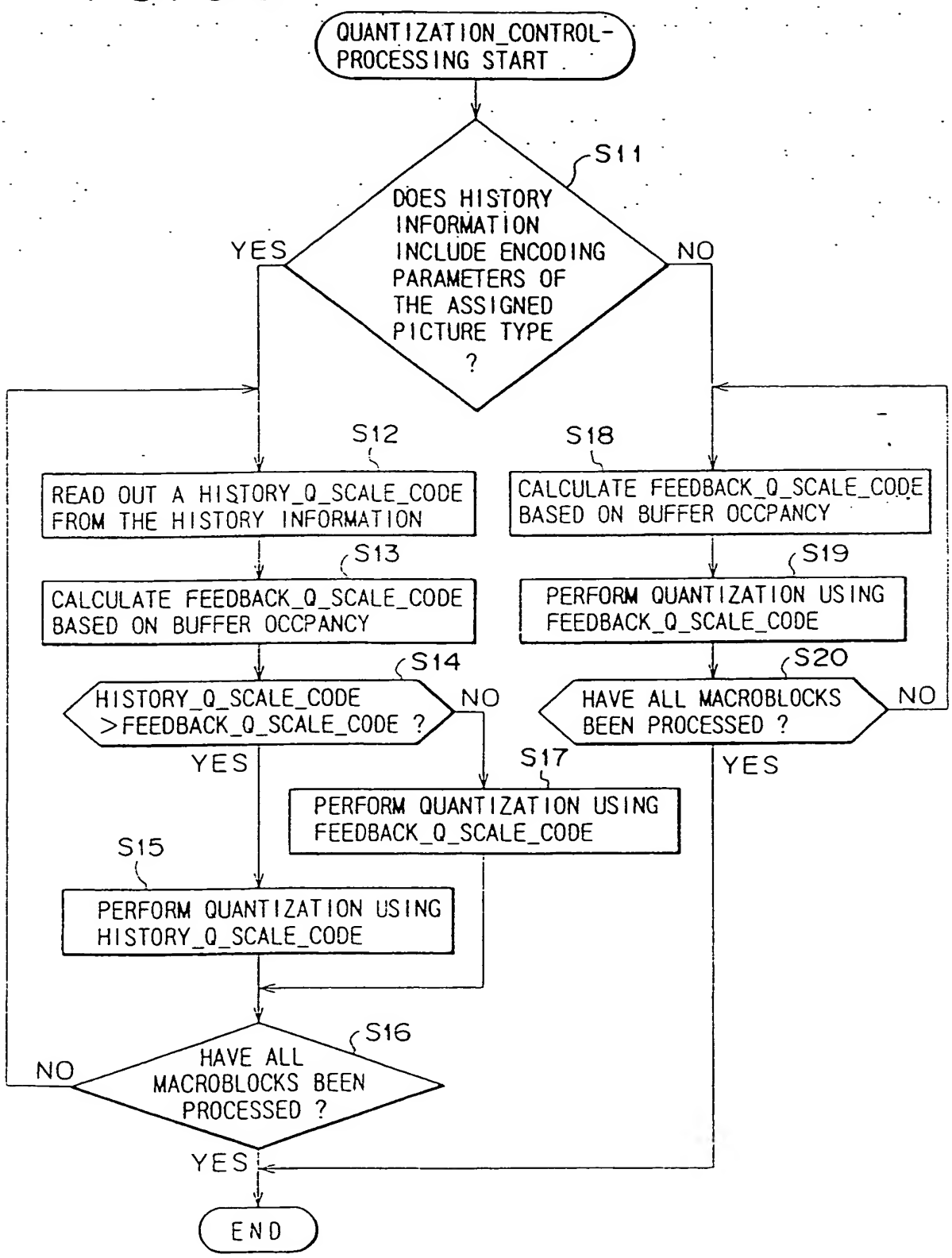

QUANTIZATION_CONTROL-PROCESSING START
S11
DOES HISTORY INFORMATION INCLUDE ENCODING PARAMETERS OF THE ASSIGNED PICTURE TYPE ?
YES
NO
S12
READ OUT A HISTORY_Q_SCALE_CODE FROM THE HISTORY INFORMATION
S13
CALCULATE FEEDBACK_Q_SCALE_CODE BASED ON BUFFER OCCPANCY
S14
HISTORY_Q_SCALE_CODE >FEEDBACK_Q_SCALE_CODE ?
NO
YES
S15
PERFORM QUANTIZATION USING HISTORY_Q_SCALE_CODE
S16
HAVE ALL MACROBLOCKS BEEN PROCESSED ?
NO
YES
S17
PERFORM QUANTIZATION USING FEEDBACK_Q_SCALE_CODE
S18
CALCULATE FEEDBACK_Q_SCALE_CODE BASED ON BUFFER OCCPANCY
S19
PERFORM QUANTIZATION USING FEEDBACK_Q_SCALE_CODE
S20
HAVE ALL MACROBLOCKS BEEN PROCESSED ?
NO
YES
END

FIG. 37

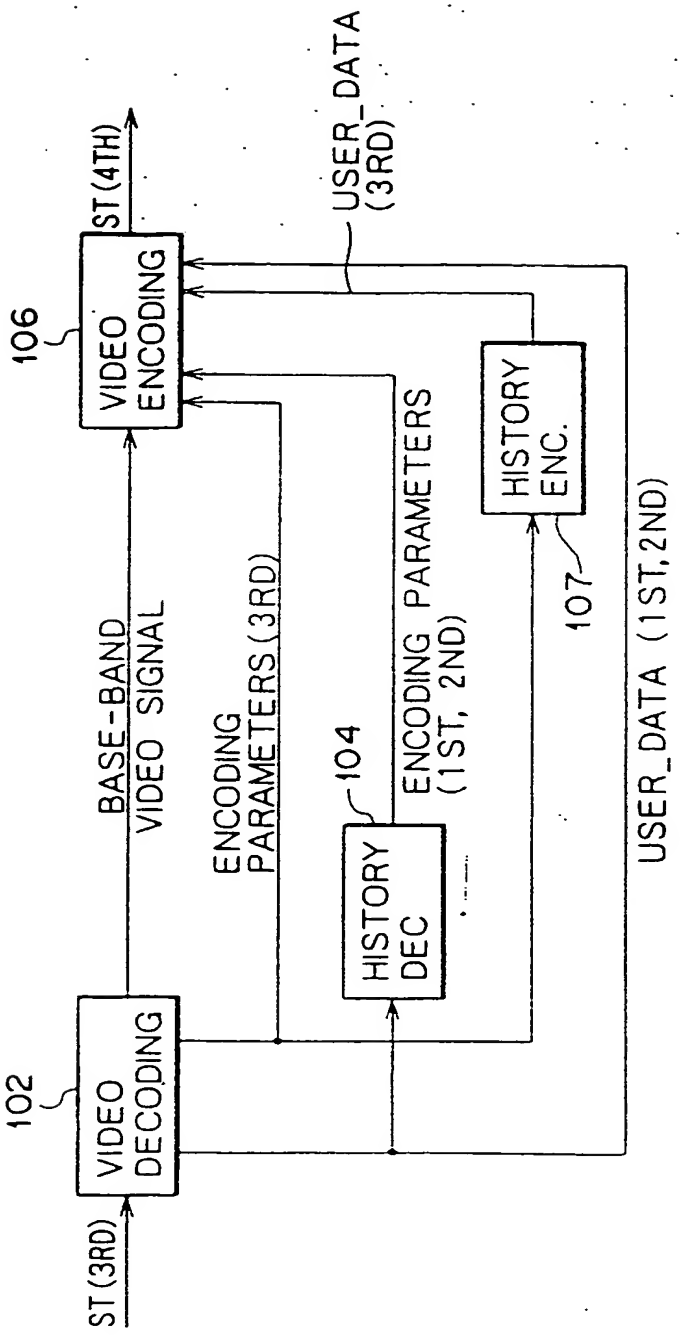

ST (3RD)
VIDEO DECODING
102
BASE-BAND VIDEO SIGNAL
VIDEO ENCODING
106
ST (4TH)
USER_DATA (3RD)
ENCODING PARAMETERS (3RD)
HISTORY DEC
104
ENCODING PARAMETERS (1ST, 2ND)
HISTORY ENC.
107
USER_DATA (1ST, 2ND)

FIG. 38

stream with history data

video_sequence( ){	No. of bits	Mnemonic
next_start_code( )		
sequence_header( )		
sequence_extension( )		
do{		
extension_and_user_data(0)		
do{		
if(nextbits( ) = = group_start_code){		
group_of_pictures_header(1)		
extension_and_user_data(1)		
}		
picture_header( )		
picture_coding_extension( )		
while((nextbits( ) = =extension_start_code)||		
(nextbits( ) = =user_data_start_code)){		
if(nextbits( ) = =extension_start_code))		
extension_data(2)		
if(nextbits( ) = =user_data_start_code){		
user_data_start_code	32	bslbf
if(nextbits( ) = =History_Data_ID){		
History_Data_ID	8	bslbf
converted_history_stream( )		
}		
else{		
user_data( )		
}		
}		
}		
picture_data( )		
}while((nextbits( ) = =picture_start_code)||		
(nextbits( ) = =groupe_start_code))		
if(nextbits( ) !=sequence_end_code){		
sequence_header( )		
sequence_extension( )		
}		
}while(nextbits( ) !=sequence_end_code)		
sequence_end_code	32	bs.bf
}		

FIG. 39

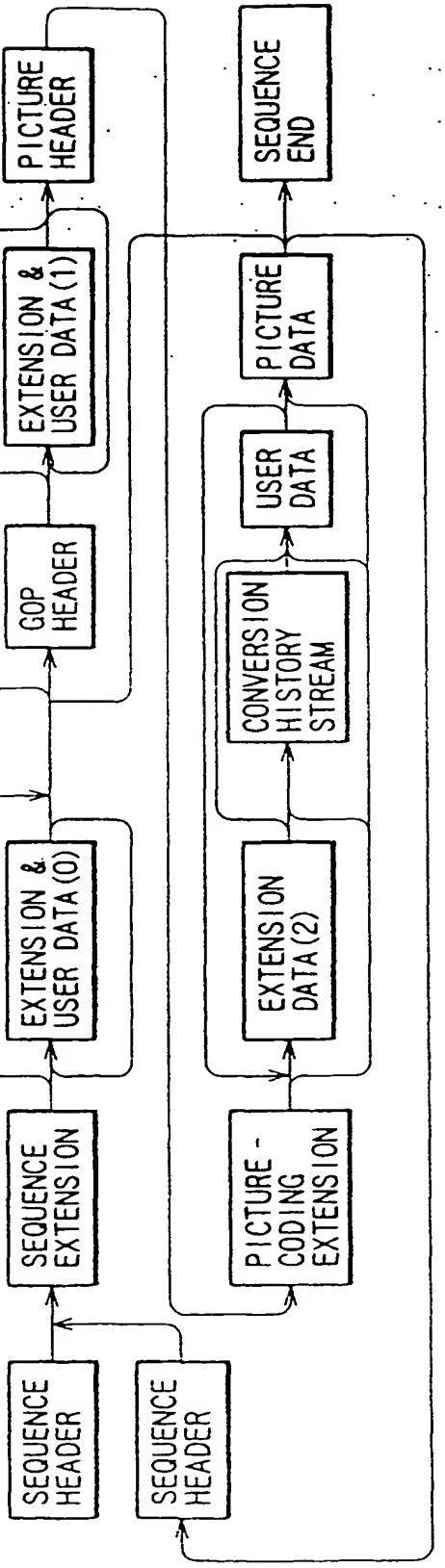

SEQUENCE HEADER
SEQUENCE HEADER
SEQUENCE EXTENSION
EXTENSION & USER DATA(0)
GOP HEADER
EXTENSION & USER DATA(1)
PICTURE HEADER
PICTURE - CODING EXTENSION
EXTENSION DATA(2)
CONVERSION HISTORY STREAM
USER DATA
PICTURE DATA
SEQUENCE END

FIG.40

history stream(40-1)

history_stream( ){	bits	value
sequence_header		
sequence_header_code	32	00000183
sequence_header_present_flag	1	
horizontal_size_value	12	
marker_bit	1	1
vertical_size_value	12	
aspect_ratio_information	4	
frame_rate_code	4	
maker_bit	1	1
bit_rate_value	18	
marker_bit	1	1
vbv_buffer_size_value	10	
constrained_parameter_flag	1	0
load_intra_quantiser_matrix	1	
load_non_intra_quantiser_matrix	1	
maker_bits	5	1F
intra_quantiser_matrix[64]	8*64	
non_intra_quantiser_matrix[64]	8*64	
sequence_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	1
sequence_extension_present_flag	1	
profile_and_level_indication	8	
progressive_sequence	1	
chroma_format	2	
horizontal_size_extension	2	
vertical_size_extension	2	
marker_bit	1	1
bit_rate_extension	12	
vbv_buffer_size_extension	8	
low_delay	1	
marker_bit	1	1

FIG. 41

history stream(40-2)

	bits	value
frame_rate_extension_n	2	
frame_rate_extension_d	5	
marker_bits	6	3F
sequence_display_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	2
sequence_display_extension_present_flag	1	
video_format	3	
colour_description	1	
colour_primaries	8	
transfer_characteristics	8	
marker_bit	1	1
matrix_coefficients	8	
display_horizontal_size	14	
marker_bit	1	1
display_vertical_size	14	
marker_bit	1	1
macroblock_assignment_in_user_data		
macroblock_assignment_present_flag	1	
marker_bits	7	7F
v_phase	8	
h_phase	8	
group_of_picture_header		
group_start_code	32	000001B8
group_of_picture_header_present_flag	1	
time_code	25	
closed_gop	1	
broken_link	1	
marker_bits	4	F
picture_header		
picture_start_code	32	00000100

FIG.42

history stream(40-3)

	bits	value
temporal_reference	10	
picture_coding_type	3	
marker_bit	1	1
vbv_delay	16	
full_pel_forward_vector	1	
forward_f_code	3	
full_pel_backward_vector	1	
marker_bit	1	1
backward_f_code	3	
marker_bit	1	1
picture_coding_extension		
extension_start_code	32	00000185
extension_start_code_identifier	4	8
f_code[0][0]	4	
f_code[0][1]	4	
f_code[1][0]	4	
f_code[1][1]	4	
intra_dc_precision	2	
picture_structure	2	
top_filed_first	1	
frame_pred_frame_dct	1	
concealment_motion_vectors	1	
q_scale_type	1	
marker_bit	1	1
intra_vlc_format	1	
alternate_scan	1	
repeat_first_field	1	
chroma_420_type	1	
progressive_frame	1	
composite_display_flag	1	
v_axis	1	
field_sequence	3	
sub_carrier	1	
burst_amplitude	7	

FIG.43

history stream(40-4)

	bits	value
marker_bit	1	1
sub_carrier_phase	8	
quant_matrix_extension		
extension_start_code	32	00000185
extension_start_code_identifier	4	3
quant_matrix_extension_present_flag	1	
load_intra_quantiser_matrix	1	
marker_bits	2	3
intra__quantiser_matrix[64]	8*64	
load_non_intra_quantiser_matrix	1	
marker_bits	7	7F
non_intra__quantiser_matrix[64]	8*64	
load_chroma_intra_quantiser_matrix	1	
marker_bits	7	7F
chorma_intra_quantiser_matrix[64]	8*64	
load_chroma_non_intra_quantiser_matrix	1	
marker_bits	7	7F
chroma_non_intra_quantiser_matrix[64]	8*64	
copyright_extension		
extension_start_code	32	00000185
extension_start_code_identifier	4	4
copyright_extension_present_flag	1	
copyright_flag	1	
copyright_identifier	8	
original_or_copy	1	
marker_bit	1	
copyright_number_1	20	
marker_bit	1	
copyright_number_2	22	
marker_bit	1	
copyright_number_3	22	3F
marker_bits	6	

FIG. 44

history stream(40-5)

	bits	value
picture_display_extension		
extension_start_code	32	00000185
extension_start_code_identifier	4	7
picture_display_extension_present_flag	1	
frame_centre_horizontal_offset_1	16	
marker_bit	1	1
frame_centre_vertical_offset_1	16	
marker_bit	1	1
frame_centre_horizontal_offset_2	16	
marker_bit	1	1
frame_centre_vertical_offset_2	16	
marker_bit	1	1
frame_centre_horizontal_offset_3	16	
marker_bit	1	1
frame_centre_vertical_offset_3	16	
marker_bits	6	3F
user_data		
user_data_start_code	32	000001B2
user_data	2048	
while(macroblock i=macroblock_count){		
macroblock		
macroblock_address_h	8	
macroblock_address_v	8	
slice_header_present_flag	1	
skipped_macroblock_flag	1	
marker_bit	1	1
macroblock_modes( )		
macroblock_quant	1	
macroblock_motion_forward	1	
macroblock_motion_backward	1	
macroblock_pattern	1	
macroblock_intra	1	

FIG.45

history stream(40-6)

	bits	value
spatial_temporal_weight_code_flag	1	
frame_motion_type	2	
field_motion_type	2	
dct_type	1	
marker_bits	2	3
quantiser_scale_code	5	
marker_bits	3	7
PMV[0][0][0]	14	
marker_bits	2	3
PMV[0][0][1]	14	
motion_vertical_field_select[0][0]	1	
marker_bit	1	1
PMV[0][1][0]	14	
marker_bits	2	3
PMV[0][1][1]	14	
motion_vertical_field_select[0][1]	1	
marker_bit	1	1
PMV[1][0][0]	14	
marker_bits	2	3
PMV[1][0][1]	14	
motion_vertical_field_select[1][0]	1	
marker_bit	1	1
PMV[1][1][0]	14	
marker_bits	2	3
PMV[1][1][1]	14	
motion_vertical_field_select[1][1]	1	
marker_bit	1	1
coded_block_pattern	12	
marker_bits	4	F
num_mv_bits	8	
num_coef_bits	14	
marker_bits	2	3

FIG.46

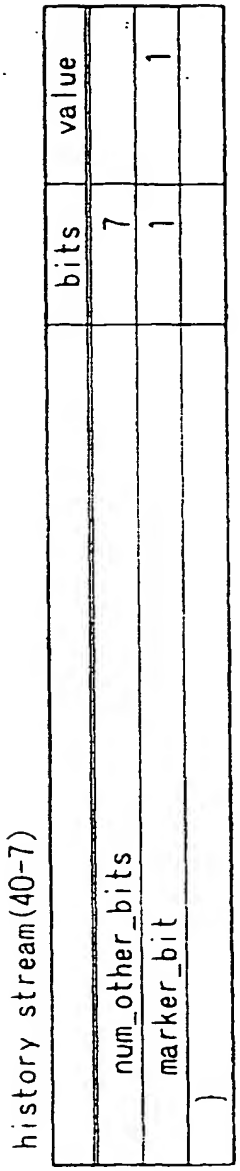


FIG. 47

history_stream( ){	No. of bits	Mnemonic
next_start_code( )		
sequence_header( )		
sequence_extension( )		
extension_and_user_data(0)		
if(nextbits( )== group_start_code){		
group_of_pictures_header( )		
extension_and_user_data(1)		
}		
picture_header( )		
picture_coding_extension( )		
extensions_and_user_data(2)		
picture_data( )		
sequence_end_code	32	bslbf
}		

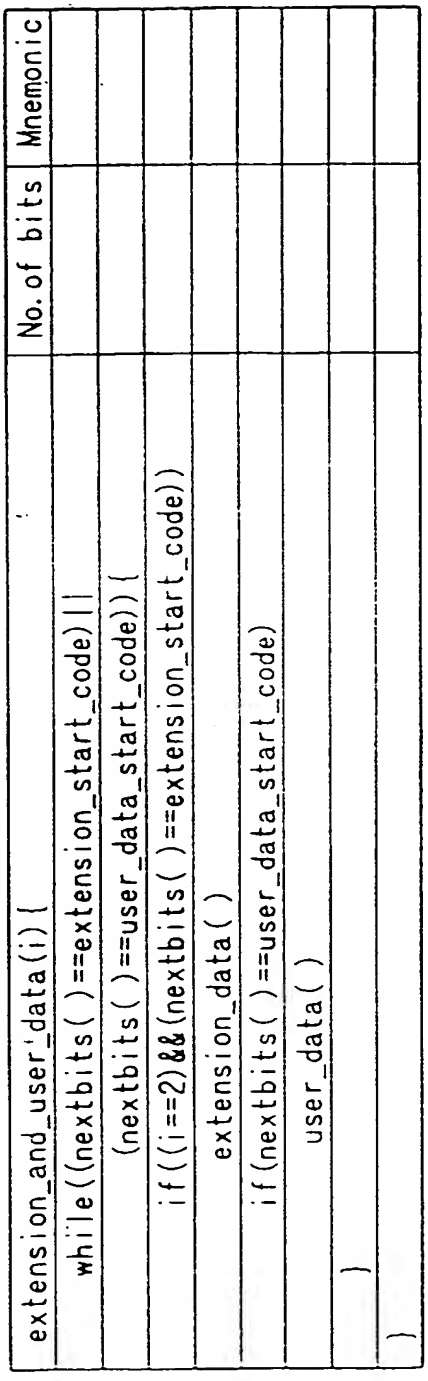
FIG. 48

sequence_header( ){	No. of bits	Mnemonic
sequence_header_code	32	bslbf
horizontal_size_value	12	uimsbf
vertical_size_value	12	uimsbf
aspect_ratio_information	4	uimsbf
frame_rate_code	4	uimsbf
bit_rate_value	18	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_value	10	uimsbf
constrained_parameters_flag	1	bslbf
load_intra_quantiser_matrix	1	uimsbf
if(load_intra_quantiser_matrix)		
intra_quantiser_matrix[64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	uimsbf
if(load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix[64]	8*64	uimsbf
next_start_code( )		
}		

FIG. 49

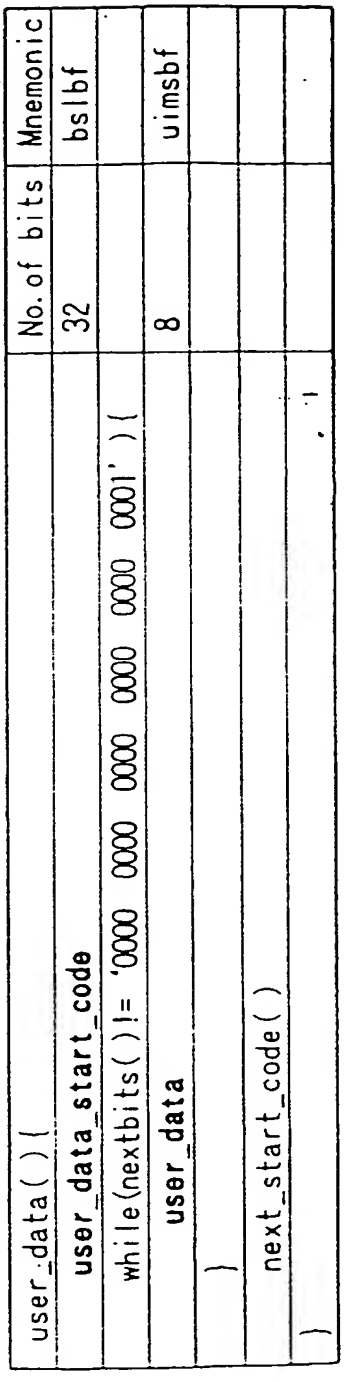
sequence_extension( ) {	No. of bits	Mnemonic
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
profile_and_level_indication	8	uimsbf
progressive_sequence	1	uimsbf
chroma_format	2	uimsbf
horizontal_size_extension	2	uimsbf
vertical_size_extension	2	uimsbf
bit_rate_extension	12	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_extension	8	uimsbf
low_delay	1	uimsbf
frame_rate_extension_n	2	uimsbf
frame_rate_extension_d	5	uimsbf
next_start_code( )		
}		

FIG. 50



extension_and_user'data(i){	No. of bits	Mnemonic
while((nextbits( )==extension_start_code)||		
(nextbits( )==user_data_start_code)){		
if((i==2)&&(nextbits( )==extension_start_code))		
extension_data( )		
if(nextbits( )==user_data_start_code)		
user_data( )		
}		
}		

FIG. 51



user_data( ){	No. of bits	Mnemonic
user_data_start_code	32	bslbf
while(nextbits( )!= '0000 0000 0000 0000 0000 0001'){		
user_data	8	uimsbf
}		
next_start_code( )		
}		

FIG. 52

group_of_pictures_header( ) {	No. of bits	Mnemonic
group_start_code	32	bslbf
time_code	25	bslbf
closed_gop	1	uimsbf
broken_link	1	uimsbf
next_start_code( )		
}		

FIG.53

picture_header( ) {	No. of bits	Mnemonic
picture_start_code	32	bslbf
temporal_reference	10	uimsbf
picture_coding_type	3	uimsbf
vbv_delay	16	uimsbf
if(picture_coding_type==2 || picture_coding_type==3){		
full_pel_forward_vector	1	bslbf
forward_f_code	3	bslbf
}		
if(picture_coding_type==3) {		
full_pel_backward_vector	1	bslbf
backward_f_code	3	bslbf
}		
while(nextbits( )=='1') {		
extra_bit_picture/*with the value'1' */	1	uimsbf
extra_information_picture	8	uimsbf
}		
extra_bit_picture/*with the value'0' */	1	uimsbf
next_start_code( )		
}		

FIG. 54

picture_coding_extension( ) {	No. of bits	Mnemonic
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
f_code [0] [0] /*forward horizontal*/	4	uimsbf
f_code [0] [1] /*forward vertical*/	4	uimsbf
f_code [1] [0] /*backward horizontal*/	4	uimsbf
f_code [1] [1] /*backward vertical*/	4	uimsbf
intra_dc_precision	2	uimsbf
picture_structure	2	uimsbf
top_field_first	1	uimsbf
frame_pred_frame_dct	1	uimsbf
concealment_motion_vectors	1	uimsbf
q_scale_type	1	uimsbf
intra_vlc_format	1	uimsbf
alternate_scan	1	uimsbf
repeat_first_field	1	uimsbf
chroma_420_type	1	uimsbf
progressive_frame	1	uimsbf
composite_display_flag	1	uimsbf
if(composite_display_flag) {		
v_axis	1	uimsbf
field_sequence	3	uimsbf
sub_carrier	1	uimsbf
burst_amplitude	7	uimsbf
sub_carrier_phase	8	uimsbf
}		
next_start_code()		
}		

FIG. 55

extension_data( ) {	No. of bits	Mnemonic
while(nextbits( )==extension_start_code) {		
extension_start_code	32	bslbf
if(nextbits( )== "Quant Matrix Extension ID")		
quant_matrix_extension( )		
else if (nextbits ( )== "Copyright Extension ID")		
copyright_extension( )		
else		
picture_display_extension( )		
}		
}		

FIG. 56

quant_matrix_extension( ) {	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
load_intra_quantiser_matrix	1	uimsbf
if(load_intra_quantiser_matrix)		
intra_quantiser_matrix [64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	uimsbf
if(load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix [64]	8*64	uimsbf
load_chroma_intra_quantiser_matrix	1	uimsbf
if(load_chroma_intra_quantiser_matrix)		
chroma_intra_quantiser_matrix [64]	8*64	uimsbf
load_chroma_non_intra_quantiser_matrix	1	uimsbf
if(load_chroma_non_intra_quantiser_matrix)		
chroma_non_intra_quantiser_matrix [64]	8*64	uimsbf
next_start_code( )		
}		

FIG. 57

copyright_extension( ){	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
copyright_flag	1	bslbf
copyright_identifier	8	uimsbf
original_or_copy	1	bslbf
reserved	7	uimsbf
marker_bit	1	bslbf
copyright_number_1	20	uimsbf
marker_bit	1	bslbf
copyright_number_2	22	uimsbf
marker_bit	1	bslbf
copyright_number_3	22	uimsbf
next_start_code( )		
}		

FIG. 58

picture_display_extension( ) {	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
for (i=0; i<number_of_frame_centre_offsets; i++) {		
frame_centre_horizontal_offset	16	simsbf
marker_bit	1	bslbf
frame_centre_vertical_offset	16	simsbf
marker_bit	1	bslbf
}		
next_start_code( )		
}		

FIG. 59

picture_data() {	No. of bits	Mnemonic
while (nextbits() == slice_start_code) {		
slice()		
}		
next_start_code()		
}		

FIG.60

slice( ){	No. of bits	Mnemonic
slice_start_code	32	bslbf
slice_quantiser_scale_code	5	uimsbf
if(nextbit( )=='1'){		
intra_slice_flag	1	bslbf
intra_slice	1	uimsbf
reserved_bits	7	uimsbf
while(nextbits( )=='1'){		
extra_bit_slice/*with the value'1'*/	1	uimsbf
extra_information_slice	8	uimsbf
}		
}		
extra_bit_slice/*with the value'0'*/	1	uimsbf
do{		
macroblock( )		
}while(nextbit( )!='000 0000 0000 0000 0000 0000')		
next_start_code( )		
}		

FIG. 63

motion_vectors(s){	No. of bits	Mnemonic
if(motion_vector_count==1){		
if((mv_format==field)&&(dmv!=1)		
motion_vertical_field_select[0][s]	1	uimsbf
motion_vector(0, s)		
}else{		
motion_vertical_field_select[0][s]	1	uimsbf
motion_vector(0, s)		
motion_vertical_field_select[1][s]	1	uimsbf
motion_vector(1, s)		
}		
}		

FIG. 64

motion_vector(r, s) {	No. of bits	Mnemonic
motion_code[r][s][0]	1-11	vlclbf
if((f_code[s][0]!=1)&&(motion_code[r][s][0]!=0))		
motion_residual[r][s][0]	1-8	uimsbf
if(dmv==1)		
dmvector[0]	1-2	vlclbf
motion_code[r][s][1]	1-11	vlclbf
if(f_code[s][1]!=1)&&(motion_code[r][s][1]!=0))		
motion_residual[r][s][1]	1-8	uimsbf
if(dmv==1)		
dmvector[1]	1-2	vlclbf
}		

FIG. 65

macroblock_type VLC code	macroblock_quant	dct_type_flag	macroblock_motion_forward	macroblock_motion_backward	Description
1	0	1	0	0	Intra
01	1	1	0	0	Intra, Quant

FIG. 66

macroblock_type VLC code	macroblock_quant	dct_type_flag	macroblock_motion_forward	macroblock_motion_backward	Description
1	0	1	1	0	MC, Coded
01	0	1	0	0	No MC, Coded
001	0	0	0	0	MC, Not Coded
0001 1	0	1	0	0	Intra
0001 0	1	1	1	0	MC, Coded, Quant
0000 1	1	1	0	0	No MC, Coded, Quant
0000 01	1	1	0	0	Intra, Quant

FIG. 67

macroblock_type VLC code	macroblock_quant	dct_type_flag	macroblock_motion_forward	macroblock_motion_backward	Description
10	0	0	0	0	Interp, Not Coded
11	0	1	1	1	Interp, Coded
010	0	0	0	0	Bwd, Not Coded
011	0	1	0	1	Bwd, Coded
0010	0	0	0	0	Fwd, Not Coded
0011	0	1	1	0	Fwd, Coded
0001 1	0	1	0	0	Intra
0001 0	1	1	1	1	Interp, Coded, Quant
0000 11	1	1	1	0	Fwd, Coded, Quant
0000 10	1	1	0	1	Bwd, Coded, Quant
0000 01	1	1	0	0	Intra, Quant